Politecnico di Torino

ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

ISTITUTO ITALIANO DI TECNOLOGIA

Doctoral Dissertation
Doctoral Program in Computer and Control Engineering (36.th cycle)

Addressing Distributional Shift challenges in Computer Vision for Real-World Applications

Francesco Cappio Borlino

* * * * *

Supervisors
Prof. Tatiana Tommasi
Prof. Barbara Caputo

Politecnico di Torino
July 16, 2024

I hereby declare that, the contents and organisation of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Francesco Cappio Borlino

Summary

Machine learning technologies have been part of our lives for a long time, as proven by widely spread applications such as automatic spam filtering and face recognition cameras. In spite of this and of the very rapid progress that characterizes research in this field, it took several years for the general public to become aware of its potential to influence our lives. However, no one can deny that this moment has now arrived, mainly due to the presentation of easy-to-use interfaces that enable anyone to interact with large language models. This development has sparked curiosity in the public about which fields *Artificial Intelligence* will influence the most. Among them, there is certainly Computer Vision, the domain in which deep neural networks have obtained the most remarkable results even before Natural Language Processing studies resulted in the development of those language models which made AI a subject on everyone's lips. The success of these models has been significantly supported by the generality of the language mean and the ease of interaction but has also led to the overestimation of their abilities, which is clearly evidenced by their inclination to make mistakes. Indeed, there is still a long way to go in order to make deep models robust enough to enable their deployment in safety-critical applications. Their brittleness gets particularly exposed when they face real-world operating conditions characterized by a large number of unforeseeable variables, as it happens when they meet out-of-distribution data. This is a situation that occurs in several scenarios, for example when a deep model faces samples with a radically different appearance from the one it is used to, or belonging to semantic categories that it has never met.

This thesis focuses on the study of these two kinds of distribution shifts. We start by providing some background, describing when they occur and why they impact so much on neural networks' performance. In this context, we focus in particular on the relationship between out-of-distribution performance and *representation learning*: the unique ability of neural networks to automatically learn how to summarize complex data samples, such as images or videos, into compact and easily tractable representations. With the goal of developing deep learning methods whose scope of applicability goes beyond lab settings, we then proceed by studying some specific distribution-shifted scenarios for which we propose novel solutions, by trying to adopt an original point of view and a critical eye on the most common paradigms. In particular, we first consider *simpler* research settings in which a visual shift is the only difference between training and

deployment conditions, and later move to more *complex* cases in which semantic and visual shifts appear together, as this is the most likely situation when considering open-world deployments.

Through our studies, we come to the conclusion that the way representations are learned can *seriously* impact the performance of deep models on out-of-distribution data, and it is thus necessary to adopt more robust learning approaches if we want to obtain dependable systems. In this context, an important novelty is represented by the recent presentation of the first *foundation models* for Computer Vision. These are models trained at scale on huge data collections that enable them to extract general-purpose representations providing a fair treatment for in-distribution and out-of-distribution data. The correct exploitation of this knowledge can thus represent a real change of paradigm in the study of distribution-shift problems.

Acknowledgements

I want to take advantage of this paragraph to thank all the people who played a part in the long process that brought me here. First of all, thank you to my supervisors, Prof. Barbara Caputo and especially Prof. Tatiana Tommasi, for guiding me step by step from the beginning of my Master's thesis to the end of my PhD one. I would like then to also express my gratitude to Prof. Nicoletta Noceti and Prof. Damien Teney for the time they spent reviewing this work, and for the valuable feedback they provided contributing significantly to its final version.

A big thank you also to my lab colleagues, in particular, Gabriele, the other Gabriele, Silvia, and Mirco, for sharing several wonderful experiences, wandering around Europe, going from big capital cities to very small villages in the far South. A special thank you also to all my coauthors with whom I shared the joys and sorrows of the scientific research life. Finally, a huge thanks to my family for always pushing me to continue studying and of course to Denise, as this was a very long journey we have traveled together, and like for any other journey I could never find a better travel partner.

Contents

Chapter 1

Introduction

1.1 Representation learning and distribution shift

The possibility to process large quantities of data, and recognize patterns that support decision-making, relieves the programmer from the task of manually designing decision rules. This technique is called **Machine Learning** and has enabled the solution of complex automation tasks that without it would be literally or practically unsolvable, by reason of the tremendous amount of required time and effort.

Deep learning has shown to be the most effective machine learning paradigm when dealing with *particularly complex* data types, like images and videos. Indeed, digital representations of these media are multidimensional matrices, with a single image being described through millions of numerical values, indicating colors for a grid of pixels. In this context, the value of a single feature, *i.e.* the intensity of a color for a specific pixel, hardly provides relevant information for the solution of the task, which, on the contrary, can be solved only by analyzing the sample as a whole. For this reason, on such complex data structures many machine learning algorithms miserably fail, even though the same algorithms provide exceptional results when they are applied to simpler structures. The success of deep neural networks, instead, is enabled by one of their key features: the ability to automatically learn to extract *compact* and *meaningful* representations from complex inputs.

Representation learning enables deep neural networks to thrive in performing complex tasks, and it is obtained for free, as a *byproduct* of their training on those same tasks. In particular, in the training phase, a *supervising* signal applied to the network's low dimensional output through an objective function is propagated back to its high dimensional input, producing an update of all of the network's parameters in the direction of making the extracted representations more and more tailored to support the solution of the training task.

This automatic representation learning ability, which many people deem the most important quality of deep learning, can, nevertheless, also become a significant obstacle

Table 1.1: Examples of visual domain and semantic shifts

	Visual domain shift	Semantic shift
Training set		
Test set		

to the implementation of deep networks in real-world systems. Indeed, when representation learning is induced by the optimization of an objective function, the learned representations may very easily start recording **supervision collapse** [36]. This phenomenon occurs when the learning signal implicitly pushes the network to extract ***only*** the features which are really relevant to reach the considered optimization objective on the considered training data. In other words, if no special precautions are taken, the network finds some *shortcuts*, ultimately learning how to represent only what is necessary to perform the task it is trained for, on the data it is trained on, disregarding any additional information which may be contained in that data.

This phenomenon becomes a problem when there is a **difference between the distributions of training and test data**, a situation possibly rare when working in lab settings, but that becomes relevant when deep models are integrated into systems deployed in the open-world. There are two main kinds of distribution's difference which commonly arise after a model's deployment (see examples in Tab. 1.1):

- **visual domain shift**. It is a **covariate shift** as it involves the input distribution (the visual appearance of input images), but not the output one (their semantic class). For example let's consider a fruit categorization problem: we want to build a system that is able to recognize three kinds of fruits. If our training dataset contains red apples and green pears, a green apple encountered after deployment would almost certainly mislead our categorization software. Another intuitive example of this situation is a change in weather conditions: if we train a pedestrian detection system for an autonomous vehicle using a training dataset of pictures collected in sunny weather, after deployment the system's performance will fall significantly any time the weather is different, even when the visibility is not impacted by its conditions;

- **semantic shift**. It is an **output distribution**'s shift: the sets of semantic classes appearing during train and test do not perfectly overlap. For example, let's consider a wildlife monitoring system: we install a camera trap in a forest and design a system to recognize three animal species for which we want to estimate the population's size. If the forest is inhabited also by other species we probably cannot prevent our trap from capturing also pictures of them. This semantic shift will

thus reduce the accuracy of our wildlife monitoring system which would confuse the unknown animals with those it has been designed to recognize.

Both these scenarios can be described as outcomes of the **presence of a bias in the training dataset** [154]. Indeed, in the fruit categorization scenario, including a larger variety of samples of apples and pears in the training dataset could enable our categorization model to understand that, in order to carry out its task, focusing on an object's shape is much more important than focusing on its color. Similarly, for what concerns the wildlife monitoring system, including in the training dataset samples of all the species that inhabit the target forest should limit the risk of encountering novel species after deployment. It is quite intuitive, however, that sometimes this kind of bias reduction effort may prove futile. Indeed, in many open-world deployment scenarios, it is impossible to cover the whole, control completely, or even limit the distribution from which test data comes from. For example, in our second case study, even if we include many more species in the problem definition, an event that's out of our control like the arrival of a new species in the area may always occur.

A bias in the training dataset is therefore **not always avoidable**, but its occurrence is not a problem as long as the test data shows the same characteristic. When this does not happen, the adoption of features automatically learned on the training data exacerbates the issue, because of the supervision collapse phenomenon. For example, going back to our first case study, this phenomenon leads the model that is only shown red apples and green pears to decide that the easiest strategy to distinguish between these two fruits is to just look at their color. As a result, all the other features of the training input samples, including the shape, are not only ignored but directly discarded in the representation learning phase. When the model is later applied after deployment, test samples are represented through feature vectors encoding only their color and nothing else. Similarly, in the wildlife monitoring case, the system is led to confuse novel with *known* categories simply because both are represented through features learned from the latter.

Building on these premises, the central research question of this thesis is **how to address the distribution shifts**. Indeed, the semantic and visual domain shifts are two consequences of the same phenomenon: the presence of a distribution shift between training and test data. As such, even some proposals of solutions are based on the same principles. For example, in the last years, a number of papers have proposed techniques to *regularize* the training, with the implicit or explicit goal of reducing the supervision collapse. With this goal in mind, some algorithms propose to adopt an auxiliary learning objective optimized jointly with the main supervised one. This procedure influences the output of the training, forcing the network to retain some additional features besides those that are useful to perform the main task. The final model is thus more robust to visual domain shifts [16]. Another strategy relies on the inclusion in the training procedure of some auxiliary samples, which do not take part in the learning of the primary task, but that, by requiring the model to be able to represent them, enable

known-unknown samples separation [51].

Despite the similarities in terms of cause and principles behind some proposed workarounds, there is a fundamental difference between the two kinds of distribution shifts on which this thesis focuses, and it is in the solutions' goal and correspondingly on the target algorithm's behaviors. In particular, dependable machine learning algorithms should provide representations that are:

- **invariant** to visual domain shifts, so that decisions are not taken on the basis of covariate variables which are not really relevant to the task at hand;
- **covariant** to the semantic shifts, so that they enable the detection of samples that do not belong to the training semantics.

The first between these two goals is the subject of study of **Domain Generalization** (DG), one of the research settings part of the broader *cross-domain analysis* research field, which includes also the **Domain Adaptation** (DA) scenario, designed for those situation in which there is some kind of *a priori* knowledge about the target visual distribution which will be met after deployment. The second goal is, instead, the problem of study of the **Out-Of-Distribution** (OOD) **detection** literature, which is strictly related to the more famous *anomaly detection* one.

The difference in treatment does not mean that the two problems are necessarily disjoint. On the contrary, they may easily occur together in real-world applications. When this happens the main risk is that strategies developed to alleviate the issues induced by one type of shift, lead to an amplification of those induced by the other one. For example, the **Open Set Domain Adaptation** (OSDA) setting studies those situations in which an unsupervised dataset coming from the target visual domain can be used to prepare a model for the visual distribution that will be met after deployment. As this dataset is unsupervised, it can contain samples that do not belong to the set of semantics of the supervised training one, samples which may make any naïve domain bridging strategy futile, leading to a phenomenon called negative transfer [93]. Similarly, if an OOD detection method is applied to test data coming from a visual distribution different from the training one, the presence of the covariate shift risks pushing the model towards marking all the test samples as belonging to novel classes [176].

This extremely diverse set of scenarios which could be met when studying a real-world problem has led to the design of a large number of research settings, some more general and some more specific, but all focusing on the tackling of a distribution shift between training and deployment data. **This thesis studies some of these settings**, describing the state-of-the-art, and proposing novel techniques to tackle their specific problems. Most of these solutions follow the standard practice of using the available training data to train from scratch or fine-tune a neural network on the task at hand, while adopting sophisticated techniques to improve the generalizability of the learned features. This approach has been the most common paradigm for a long time as it guarantees to obtain models that perform well at least on the in-distribution data, even if the

corresponding performance on out-of-distribution data may be poor because of supervision collapse. In the very last years, the factors that have long made this approach the most valid and successful have started fading out. The cause of this phenomenon can be found in the significant progress made in the development of strategies for large-scale training of neural networks, which in turn has been made possible by the development of self-supervised learning techniques that enable learning on huge data collections. After the successes achieved in the field of Natural Language Processing, these improvements have led to the presentation of the first ***foundation models*** also in the Computer Vision world. These are models that, in reason of the scale of their training, are able to provide high-quality general-purpose representations ready to be used for downstream tasks [9]. Their development has the potential to force a paradigm shift in many CV research fields and in particular in those fields where feature generalizability is fundamental, as is clearly the case of distribution shift analysis. In this area, the availability of general-purpose features able to accurately and fairly represent both in-distribution and out-of-distribution data may be sufficient grounds for completely discarding networks' training or fine-tuning on in-distribution data, a procedure which can easily hurt any previously learned knowledge. The last contribution of this thesis is thus the first analysis of the impact that a correct use of foundation models has on distribution shift analysis, focused in particular on assessing the performance that these models can unlock if compared to traditional strategies.

1.2 Contributions

This work studies the visual domain and semantic distribution shifts from a set of different points of view. Novel solutions are presented and large-scale experimental evaluations are carried out to draw a comprehensive picture of the state-of-the-art and to better understand the advantages and disadvantages of different algorithms. The main contributions of this thesis are:

- the presentation of the **first strategy to adapt an object detector on a single test sample** (Chapter 3). The proposed approach exploits a self-supervised task to enable training on unsupervised data in order to adapt the detector's backbone to the target's distribution. We also provide a proof-of-concept of how meta-learning can help in this context by making adaptation faster;
- the proposal of a **novel robust baseline for Domain Generalization** (Chapter 4) exploiting style-transfer as part of a data augmentation pipeline that enables obtaining domain-invariant features;
- the presentation of the **first approach** able to tackle all the challenges of the complex **multi-source open-set domain adaptation** setting with a **single learning objective** (Chapter 5);

- the proposal of a similar algorithm adapted to the **cross-domain open-world recognition** task by the introduction of a class-incremental learning component (Chapter 5);
- the **introduction and proof-of-concept** of a tailored representation learning strategy specifically designed to support performing OOD detection on a broad range of tasks without fine-tuning (Chapter 6). The proposed approach is based on **relational reasoning** and enables a model to provide a semantic similarity measure on pairs of pictures;
- the **definition of a novel, comprehensive benchmark for the OOD detection task** (Chapter 6), overcoming the problems in terms of limited scale and low transferability to real-world scenarios of previous benchmarks, and including both an intra-domain and a cross-domain track;
- the first large-scale **experimental comparison of fine-tuning-free and fine-tuning-based OOD detection** approaches (Chapter 6), studying for the first time also the **impact that *foundation models*** could have on future research developments in the setting.

All the algorithms proposed in this thesis are thoroughly analyzed through comprehensive testbeds, which are inherited from the literature when they are both relevant to the studied problem and realistic in terms of transferability of the results to real-world scenarios, or designed from scratch to have these characteristics when literature testbeds do not match these constraints.

1.3 Thesis outline

The **second chapter** of this thesis provides a **general overview** of the two distribution shift types on which we focus. The chapter introduces a formal definition of the most general research settings tackling visual domain or semantic shift, and a summary of the related research works. In this way we offer the reader some basis about both the general problems in analysis and the solutions that have been proposed to deal with them, in order to ease the understanding of the subsequent chapters which consider also more specific settings and dive in more details about proposed solutions.

In the **third chapter**, we focus on **one-shot unsupervised cross-domain detection**, a research setting designed to study those real-world scenarios in which an object detector faces a continuously varying visual distribution after deployment, as is the case of social media monitoring applications. In this case, traditional domain adaptation solutions are unsuited to provide predictions tailored for the ever-changing visual domain and a strategy able to adapt on a single test sample should be preferred. The proposed

solution adopts self-supervision to reach adaptation of a neural network backbone before providing predictions. The chapter also presents an extension of this initial solution which exploits meta-learning to increase adaptation speed, and provides a comprehensive experimental section comparing the proposed strategies with state-of-the-art domain adaptation solutions.

The **fourth chapter** focuses on the general **Domain Generalization** setting, analyzing its **relationship with data augmentation**. Indeed, this broad research setting has seen the proposal of a large number of algorithms, based on widely different strategies, some of them particularly sophisticated but also rather complex. Most of these approaches have the common goal of training models to provide domain-invariant features but often neglect the contribution that data augmentation can have in this context. The chapter thus proposes a very simple style-transfer-based data augmentation pipeline that allows obtaining a robust DG baseline that outperforms previous state-of-the-art approaches. We analyze how this data augmentation strategy can be combined with previous solutions aiming to raise attention to the fact that many state-of-the-art methods do not provide any advantage when they are combined with this improved baseline. This finding should push for the development of novel DG strategies able to take advantage of the proposed data augmentation pipeline in order to build even more robust algorithms.

The two settings studied in the **fifth chapter** are **multi-source open-set domain adaptation** and **cross-domain open-world recognition**. The link between them is that they are two instances of open-world learning problems in which multiple challenges are faced at the same time. The two proposed solutions are thus equally connected by being based on the same principle: the use of a single contrastive-based learning objective that allows to tackle multiple challenges jointly thanks to the structural properties of the learned feature space. We discuss the advantages of these solutions with respect to competitors that, in the majority of cases, are obtained as combinations of strategies designed for individual sub-problems of the main one.

In the **sixth chapter**, a **paradigm shift** is proposed for the tackling of the **Out-Of-Distribution detection** problem. In this context, the standard practice consists in performing a complete training, or at least a fine-tuning of a pre-trained model, on the in-distribution data of the task at hand. This procedure has a number of disadvantages, ranging from the poor out-of-distribution representation capabilities induced by supervision collapse and forgetting of the pre-trained knowledge, to the high computational cost of performing the needed training session. In order to overcome these disadvantages, we propose to completely discard the fine-tuning phase by relying only on pre-trained knowledge to build comparison strategies between in-distribution *support* data and test samples, with the final goal of providing normality scores for the latter. In particular, we propose a relational reasoning-based representation learning paradigm designed to learn a generic semantic similarity measure through training on

a large dataset. For this model, we perform a thorough analysis of alternative learning objectives and then we frame it in the bigger picture of a large-scale experimental comparison between fine-tuning-free and fine-tuning-based OOD detection approaches. In this context, we consider a wide number of pre-training solutions in order to assess their applicability to the studied problem and we focus in particular on *foundation models* because of the impact that they can have on the whole research field.

In the **seventh** and last chapter, the focus is on drawing conclusions from the whole work, in order to better outline the outcome of the conducted analyses and describe possible future research directions.

1.4 Publication list

We list here the publications of the author in chronological order. When they are public we also include links to repositories containing the codebases implementing the proposed algorithms and enabling experiment reproduction. Some of the listed publications, highlighted with a (*), present contents not included in this thesis:

- [33] A. D'Innocente, F. Cappio Borlino, S. Bucci, B. Caputo, and T. Tommasi
  One-shot unsupervised cross-domain detection
  European Conference on Computer Vision, ECCV 2020
  Code: https://github.com/VeloDC/oshot_detection

- [19] F. Cappio Borlino, A. D'Innocente, and T. Tommasi
  Rethinking Domain Generalization Baselines
  25th International Conference on Pattern Recognition, ICPR 2020

- [21] F. Cappio Borlino, S. Polizzotto, B. Caputo, and T. Tommasi
  Self-Supervision & Meta-Learning for One-Shot Unsupervised Cross-Domain Detection
  Computer Vision and Image Understanding Journal (CVIU), 2022
  Code: https://github.com/FrancescoCappio/OSHOT-meta-learning

- [15] S. Bucci, F. Cappio Borlino, B. Caputo and T. Tommasi
  Distance-based Hyperspherical Classification for Multi-source Open-Set Domain Adaptation
  Winter Conference on Applications of Computer Vision, WACV 2022
  Code: https://github.com/silvia1993/HyMOS

- [17] F. Cappio Borlino, S. Bucci, and T. Tommasi
  Contrastive Learning for Cross-Domain Open World Recognition
  The 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2022)
  Code: https://github.com/FrancescoCappio/Contrastive_Open_World

- (*) [3] A. Alliegro, F. Cappio Borlino, and T. Tommasi
  3DOS: Towards 3D Open Set Learning –
  Benchmarking and Understanding Semantic Novelty Detection on Point Clouds
  Proceedings of the Neural Information Processing Systems (NeurIPS)
  Track on Datasets and Benchmarks, 2022
  Code: https://github.com/antoalli/3D_OS

- [18] F. Cappio Borlino, S. Bucci, and T. Tommasi
  Semantic Novelty Detection via Relational Reasoning
  European Conference on Computer Vision, ECCV 2022
  Code: https://github.com/FrancescoCappio/ReSeND

- [101] L. L. Lu, G. D'Ascenzi, F. Cappio Borlino, and T. Tommasi
  Large Class Separation is not what you need for Relational Reasoning-based OOD Detection
  International Conference on Image Analysis and Processing, ICIAP 2023
  Code: https://github.com/lulor/ood-class-separation

- [20] L. L. Lu, F. Cappio Borlino, and T. Tommasi
  Foundation Models and Fine-Tuning: A Benchmark for Out Of Distribution Detection
  IEEE Access, 2024
  Code: https://ooddb.github.io/
  https://github.com/FrancescoCappio/OODDetectionBench

Chapter 2

Background and related works

The purpose of this chapter is to give an overview of the two distribution shift problems studied in this thesis, by providing both a **formal definition** of their most common formulations and some **background** on the literature focusing on them. Both the *visual domain* and the *semantic* distribution shifts represent significantly wide research fields which include a broad range of specific research sub-settings. As a result, this chapter will present only the definition and background for the *general* versions of these problems, whereas the next chapters will mainly focus on *specific* scenarios and will thus include a formal definition and some background for them.

2.1 Preliminaries

We consider learning problems in which the goal is to learn a mapping $y = f(\boldsymbol{x})$ from a dataset of pairs $(\boldsymbol{x}, y)$, where $\boldsymbol{x} \in \mathcal{X}$, $y \in \mathcal{Y}$ and the pairs are sampled from the joint distribution $p(\boldsymbol{x}, y)$. A distribution shift between training and test (*i.e.* deployment) data happens when $p_{training}(\boldsymbol{x}, y) \neq p_{test}(\boldsymbol{x}, y)$.

2.2 Visual domain shift

A visual domain shift is a covariate shift, which means that $p_{training}(\boldsymbol{x}, y) \neq p_{test}(\boldsymbol{x}, y)$ because $p_{training}(\boldsymbol{x}) \neq p_{test}(\boldsymbol{x})$ even though $p_{training}(y|\boldsymbol{x}) = p_{test}(y|\boldsymbol{x})$. This kind of shift happens when there is a difference in the input distribution involving visual features that do not influence the semantics. Considering real-world applications this can happen for a multitude of reasons.

For example, considering a widely known application such as autonomous driving, it is virtually impossible to foresee all the visual scenarios that the model will face after deployment. Indeed, in this case, visual differences may go from variations in the lighting conditions to variations in weather, from the differences between city streets and country roads to the diverse architectures that could be met when traveling through different countries. In this case, in order to avoid domain shifts, all the combinations of all these circumstances should be taken into account when collecting the training, which makes this task highly complex and particularly expensive, if not actually unfeasible.

Another common situation occurs when training is performed by choice on a dataset not specifically designed and collected for the target task. There could be many reasons to do so: the target domain could not be known *a priori*, or collecting and or labeling data from it could be too expensive. This last aspect is often not negligible, given the amount of data necessary to train neural networks, and for this reason there are research lines focusing specifically on how to reduce the labeling costs [145, 124]. When this cost is too high parties sometimes focus only on data collection, which leads to the construction of unsupervised target datasets.

One last example of very common domain shift involves all those applications for which the training is performed on synthetic data, but the deployment is still performed on real data. This is the case for example of many robotics scenarios [59], but it happens often also anytime learning has to be performed on data types for which large real-world data collections do not exist yet, for example 3D point clouds [3].

A domain shift is thus a situation much more common than one could initially think, and the severity of the issues that it causes could of course vary depending on the application, but it is rarely negligible. Indeed, unless otherwise guided, neural networks yield features that describe mostly local rather than global patterns [185, 58], which means that they place great emphasis on characteristics that are not consistent across domains.

Many research settings have been designed in order to analyze domain-shifted scenarios and propose solutions and strategies to deal with this problem. We can organize literature studies into two main big families: *Domain Generalization*, and *Domain Adaptation*. The difference among them relies on the possibility of accessing at least some data from the target visual domain at training time.

2.2.1 Domain generalization

The Domain Generalization (DG) setting has been designed to analyze the robustness of deep learning models across visual domains. In this setting, models are trained on one or more labeled *source* datasets, which are collected by sampling from distributions corresponding to different visual domains. The models' performance is later evaluated on one or more *target* datasets corresponding to previously unseen domains. A robust model proves generalization ability by making decisions avoiding any influence from domain-specific features, which are irrelevant to the task. This ability may be innate if it comes from the structure of the model itself or may be developed through specific training strategies, like the adoption of particular data augmentation approaches or auxiliary learning objectives.

The most common DG development and evaluation protocol relies on the use of multiple *source* domains and a single *target* one. Indeed, the availability of multiple labeled source datasets corresponding to different visual domains enables a better understanding of which features are domain-specific and which are domain-invariant. The exclusive use of domain-invariant features guarantees that the model will later provide consistent results across any domain it may meet after deployment.

The next sections provide a formal definition of the problem followed by a short summary of the literature studying it.

Problem formalization

We formally describe the problem in terms of **data available at training time**, **task goal**, and **data available at test time**. This **problem formalization framework** is both formal enough to provide in an unambiguous way all the necessary information to completely describe a research problem, and versatile enough to be applied to all the problems studied in this thesis.
Problem formalization:

- **data available at training time**: n_S labeled source datasets $S = \{S_i\}_{i=1}^{n_S}$, each one consisting of a set of image-label pairs: $S_i = \left\{ \left(\boldsymbol{x}_j^{(s)}, y_j^{(s)} \right) \right\}_{j=1}^{N_{S_i}}$. In this context $\boldsymbol{x} \in \mathcal{X}$ is a multidimensional input (an image) coming from the input space $\mathcal{X}$, while $y \in \mathcal{Y}$ is the ground-truth label coming from the label space $\mathcal{Y}$

- **goal**: to correctly classify samples in the test set $T = \{(\boldsymbol{x}_i^{(t)}, y_j^{(t)})\}_{i=1}^{N_T}$. We have $\mathcal{Y}_S = \mathcal{Y}_T$ and all the datasets are sampled from different marginal distributions and thus show different visual characteristics;
- **data available at inference time**: $\{\boldsymbol{x}_i^{(t)}\}_{i=1}^{N_T}$

In practice, in DG we want to build a model which exploits the available data sources to learn a mapping $y = f(\boldsymbol{x})$ which generalizes to any target visual domain. There are two main DG versions: the most common is **multi-source DG**, where $n_S > 1$, but in some scenarios a single source dataset is available ($n_S = 1$) and thus it is important also to study **single-source DG** strategies.

Related works

The domain generalization task has recorded a significant interest increase in the research community in the last years, which has led to the proposal of a wide variety of approaches to deal with it. Existing methods can be roughly divided into three main groups.

Feature Alignment These approaches inherit from one of the most common strategies adopted in domain adaptation literature, which consists of measuring the distance between domains and learning a representation that minimizes it. In the DG setting, this strategy is applied among the available sources through MMD discrepancy constraints [84], metric learning (contrastive loss) [114], or by adopting adversarial domain classifiers [86].

Meta-Learning Solutions of this group divide the sources into two groups called meta-train and meta-test: a model is learned on the former with the real goal of reducing the error on the latter, often exploiting multiple learning episodes. In this way, it is possible to prepare the model for the domain shift that will be experienced on the actual target. Two of the most well-known approaches exploit episodic training with [83], or without [82] an ad-hoc second-order gradient descent update rule inspired by MAML [39].

Self-supervision The adoption of an auxiliary self-supervised task has recently been shown to support generalization by regularizing the model's training. In [23] the jigsaw puzzle task is used as an auxiliary learning objective together with supervised object classification, helping the model to focus on the object parts and their shape rather than on domain-specific features, such as the texture. Another solution [172] exploits self-supervised rotation recognition with a similar objective. Before self-supervision, unsupervised learning already demonstrated a beneficial effect on generalization, for example through clustering [106].

Alternative solutions Other approaches, which do not fall in the above categories, of course, exist. For example, a line of research proposes to train domain-specific entire networks or sub-modules, and then weight their prediction output according to the similarity between the test sample and the source domains [32]. Other works try to explicitly disentangle domain-specific and domain-invariant features [126], in order to focus on the latter for robust learning. A possible strategy is also to focus on augmenting the source data, by exploiting pixel-level transformation, with the aim to train models that are robust to these transformations [189].

2.2.2 Domain adaptation

The Domain Adaptation (DA) research setting has been designed to study all those situations in which the target domain is known *a priori*, and some target samples are available during training. Still, the access to the target domain is limited and thus it's not possible to freely collect a large supervised dataset from it. Many variations of this scenario can be imagined and for this reason there exists an equally high number of research sub-settings dealing with it.

For example, in *supervised* DA a small set of target samples are actually labeled and this supervision can be used together with a large amount of unlabeled target samples and of labeled source ones to train a model suited to operate in the target environment.

The most common research sub-setting, which is also the most general and thus the one on which we focus here, is, however, the *Unsupervised Domain Adaptation* (UDA) one. In this case, no supervision exists for the target samples available at training time. A labeled source dataset is thus used to train a model so that it learns the task concepts, and this knowledge is then adapted in some way to the target domain. Indeed, even if they are not supervised, target samples provide plenty of information that can be used by UDA algorithms to try to prevent or bridge any representational *misalignment* between source and target data. This kind of misalignment is the main cause of the performance drop which is recorded by models when they operate in the presence of a domain shift: because the source and target samples are mapped in different regions of the space, the decision boundaries learned on source data do not align correctly with target data. As a result, by improving the domain alignment it is possible to limit the performance drop.

Problem formalization

The UDA setting is characterized by:

- **data available at training time**:
  - a source dataset composed of image-label pairs $S = \left\{ \left(\boldsymbol{x}_j^{(s)}, y_j^{(s)} \right) \right\}_{i=1}^{N_S}$, where $\boldsymbol{x} \in \mathcal{X}$ is a multidimensional input (an image) coming from the input space $\mathcal{X}$, while $y \in \mathcal{Y}$ is the ground-truth label coming from the label space $\mathcal{Y}$;

  - an unlabeled target dataset $T_1 = \left\{ \boldsymbol{x}_j^{(t_1)} \right\}_{j=1}^{N_{T_1}}$

- **goal**: to correctly classify samples in the test set $T_2 = \{(\boldsymbol{x}_i^{(t_2)}, y_i^{(t_2)})\}_{i=1}^{N_{T_2}}$. We have $\mathcal{Y}_S = \mathcal{Y}_T$ and we assume that T_1 and T_2 are sampled from the same distribution p_T, which is different from p_S from where S comes;

- **data available at inference time**: $\{\boldsymbol{x}_i^{(t_2)}\}_{i=1}^{N_{T_2}}$

Many research benchmarks are built on the *transductive* case, where $T_1 = T_2$. In practice, in this situation, a large unlabeled target dataset is trained to adapt a model to the target domain before performing predictions on the **same** target dataset. Most real-world applications however are *non-transductive*, which means that the knowledge learned from the target dataset T_1, which is available at training time, needs to be applied to novel target data T_2 after deployment.

In most cases, the UDA problem is tackled in a *single-source* scenario, but a *multi-source* one is equally possible and clearly guarantees access to a potentially broader knowledge base, even if its transfer into a single model shows its own set of challenges [127].

Even if the situation $\mathcal{Y}_S = \mathcal{Y}_T$ is the most common, some settings consider the possibility that there may not be a perfect overlap between source and target label sets. Indeed, guaranteeing this perfect overlap may not always be possible given the unsupervised nature of target data. A number of sub-settings study situations in which this overlap is not guaranteed, this is the case, for example, of *Universal* DA [42], and *Open-set* DA [123].

Related works

The final goal of most, if not all, DA approaches consists in *bridging* in some way the distribution discrepancy between source and target data. The reason to do so is that a famous theoretical analysis of the problem has led to the definition of an **error bound** in cross-domain tasks as the sum of the source error with a measure of divergence between the two domains [4]. This domain bridging can be obtained in different ways, we organize literature methods in four main groups.

Domain distance and alignment A first family of algorithms proposes approaches to estimate the distance between the two distributions and strategies to minimize it during training. In [137], Saito *et al.* exploit the $\mathcal{H}\Delta\mathcal{H}$ divergence proposed in [4] in order to minimize the discrepancy between two classifiers. When focusing directly on the features, some methods use the kernel Maximum Mean Discrepancy (MMD) [11] as a distance measure between the domains and try to minimize it [100], while others use kernel-based metrics to improve alignment [45], or try to directly match the two distributions by minimizing the differences in terms of mean and covariance [146].

Adversarial alignment A significantly large line of research focuses on improving feature alignment without relying on explicit distance metrics, but by exploiting the adversarial learning principle developed for Generative Adversarial Networks (GANs) [46]. The base approach, in this context, consists in using a domain discriminator trained adversarially with the network's feature extractor. This result can be obtained by using a simple gradient reversal layer between the feature extractor and the domain discriminator [43]. In [155], Tzeng *et al.* propose a more complex approach that uses domain-specific networks and a domain discriminator to align their output, later processed by a classifier. CDAN [99] uses two conditioning strategies to improve the adversarial alignment.

Pixel level adaptation A family of strategies focuses on obtaining the alignment at the pixel-level instead of the feature-level. Similarly to what happened with adversarial alignment methods, even approaches that focus on pixel-level adaptation exploit the capabilities of GANs to reach their goal. Both SBADA-GAN [135] and Cycada [52] use a bidirectional GAN similar to CycleGAN to transfer source images to the target domain, where they can be used to train a target model. An inverse transfer is also used to make sure that the overall transformation procedure preserves the semantics. On its hand [12] relies on a one-way only transformation.

Regularization A number of other methods obtain the domain bridging effect with more indirect approaches, generally involving some changes to the learning objective or the introduction of auxiliary ones. A group of them exploits self-supervision, usually implemented as part of a joint learning objective with a supervised task [16, 173]. The main advantage of this approach is that, even if it does not force alignment directly, self-supervision can be applied in the same way on supervised and unsupervised data, thus leading the model to learn coherent features between the two. An analogous quality also characterizes entropy-based learning objectives, which have been thus applied for domain adaptation as well [103]. We can include in this group also MCC [62], which focuses on modeling the class confusion and tries to minimize it. Another regularization approach is presented in AFN [175], which, on its part, noted that target data tend to have features with significantly smaller norms than source data. This difference arises from the fact that the supervised learning objective pushes the norm of supervised samples' representations to grow indefinitely while the training proceeds. An improved alignment can thus be obtained by adopting a learning objective which progressively adapts the feature norms of the two domains to a large range of value.

2.3 Semantic Distribution shift

A semantic distribution shift is a distribution shift which involves the output distribution. In this case $p_{training}(\boldsymbol{x}, y) \neq p_{test}(\boldsymbol{x}, y)$ because $p_{training}(y) \neq p_{test}(y)$. Given that we

focus on problems in which the output is connected to semantics when dealing with this problem we often describe it as a semantic *misalignment* whose strength can vary significantly: it can happen that the set of semantics included in the train and test distributions do not overlap, that the test distribution includes outliers or even very small variations of the semantics appearing in the training one.

Real-world examples of these situations are the encounter by an autonomous vehicle of an unidentified object on the road, the presence on a monitored industrial conveyor belt of an anomalous object, or the occurrence of defects on products checked for quality assurance. The differences among these situations may seem significant, because the examples of semantic shifts, that we have highlighted, clearly have different strengths, and because the safety risks associated with these situations are not comparable. At the same time, from the point of view of formal problem definitions, the discrepancies are minimal: there is a closed and clearly defined set of semantics that represents the *normality*, and a dependable machine learning model that should be able to detect everything that deviates from it. This last point may also be the most important connection among the described situations: in most semantic shift scenarios, while the *normality* is clearly defined, the *abnormality* is not, as it simply corresponds to the opposition to the normality. This is both a forced and a practical choice: indeed, it is usually impossible to foresee all the semantic variations that could characterize anomalous samples, and thus it is not possible to collect training data for all of them. As a result, it is much more practical to rely on collecting only samples of *normal* data, and then try to model the distribution from which they come, in order to be able to detect deviations from it.

Despite these similarities, there exist various research settings studying the semantic distribution shift from different points of view. Some examples are [177]:

- **Out-Of-Distribution** (OOD) **detection**. Although its name clearly describes a significantly broad research problem, in its most common formulation it is used to refer to a scenario with a much stricter scope, *i.e.* the task of detecting samples belonging to any semantic category not appearing in the closed-set of *known* ones;

- **Open-set Recognition** (OSR). It is similar to the previous setting, but with the additional requirement of correctly classifying all samples belonging to *known* categories;

- **Anomaly Detection** (AD). It is very similar to OOD detection, with the slight difference that generally the *normality* is treated explicitly as homogeneous, as if it included a single semantic concept. For example, in industrial defects monitoring, anomaly detection models are usually trained on a single category of products (i.e. samples of a specific product) and, at test time, they analyze samples that belong to the same semantic category, looking for discrepancies that could highlight defects. There is also a difference in terms of *motivation* behind these tasks:

in AD, the anomalous samples are usually perceived as erroneous, fraudulent or malicious, while in OOD detection they are simply considered *unknown*.

Among these sub-problems, we focus on the first one because of its generality.

2.3.1 Out-Of-Distribution detection

Most machine learning algorithms, in order to work as expected, need to be applied to data that respects the so-called *iid assumption*, which means that the data samples should be **independent and identically distributed**. In many real-world applications, however, this condition cannot be enforced and is, in practice, often unfulfilled. One of the phenomena that could lead to this situation is the occurrence of out-of-distribution samples among the ones that the model meets after deployment. As the name suggests, these samples belong to a different distribution w.r.t. the training ones and could cause a serious drop in the model's performance. Of course, this distribution shift can be of any type, *i.e.* a covariate (like the visual domain shift) or a semantic one.

In most cases, however, the name **OOD detection** is used to refer to the detection of the second type [177]. The reason may be linked to the different expectations we have of what a robust autonomous agent should do when dealing with one of these two issues, as anticipated in Chapter 1. Indeed, in the majority of cases, we want a robust model to be **invariant** to covariate shifts, while it should **detect** the semantic ones.

Problem formalization

Considering the usual problem formalization framework:

- **data available at training time**: a *support set* $S = \{(\boldsymbol{x}_i^{(s)}, y_i^{(s)})\}_{i=1}^{N_S}$ of sample-label pairs, where $y_i^{(s)} \in \mathcal{Y}_S$, and $\mathcal{Y}_S$ identifies the *known* class set. The samples belonging to it are considered in-distribution (ID);
- **goal**: to correctly binary-classify the samples of the test set $T = \{(\boldsymbol{x}_j^{(t)}, y_j^{(t)})\}_{j=1}^{N_T}$ in one of the classes in $\mathcal{Y}_K = \{known, unknown\}$, where $y_{k,i} = known$ if $y_i \in \mathcal{Y}_S$. We have $y_j^{(t)} \in \mathcal{Y}_T$, and $\mathcal{Y}_S \subset \mathcal{Y}_T$. Samples belonging to $\mathcal{Y}_{T \setminus S} = \mathcal{Y}_T \setminus \mathcal{Y}_S$ are considered out-of-distribution (OOD) and should be classified as *unknown*;
- **data available at inference time**: $\{\boldsymbol{x}_j^{(t)}\}_{j=1}^{N_T}$

The *support* set defines the *normality*, hence the purpose of an OOD detector is to detect test samples that deviate from this normality, by marking them as *unknown*. In practice, this means that the detector provides for each test sample a *normality score*, *i.e.* a scalar value which should allow to discriminate known test samples from unknown ones by imposing a threshold on it. In most cases, the definition of the threshold, which is a quite

complex and tricky problem, is not considered part of the design of an OOD detector. Indeed, the choice about the threshold should depend on the application, which should define if it is more important to avoid False Positives or False Negatives (see definitions in section A.3.2).

Related works

We group literature methods designed for Out-Of-Distribution detection in four categories.

Discriminative Methods It's common to believe, that a model trained for object recognition on a supervised dataset will then provide predictions with high confidence for in-distribution data, and high uncertainty when facing OOD samples. As a result, it should be possible to separate known and unknown test samples by simply using the classification confidence as normality score. This is the approach exploited by the most common OOD baseline [50] which uses the maximum softmax probability (MSP) as normality metric. This approach, however, is not particularly robust, as deep neural networks suffer from *overconfidence* [118], which means that they tend to provide predictions with high confidence even for images not related to their training distribution. Various approaches have been thus proposed to re-calibrate networks' outputs. In [158] Vaze *et al.* propose to discard the normalization effect of the softmax function and use directly the maximum logit score (MLS) for estimating normality. ODIN [90] exploits temperature scaling and input pre-processing to improve known-unknown separation. An alternative consists in using Energy [95] scores instead of the original prediction output, as these prove to be better aligned with the probability density function of the inputs. With GradNorm [53], Huang *et al.* proposed to focus on the network's gradients, arguing that their norm enables distinguishing which samples are far from the known distribution. ReAct [148] uses a rectification (clipping) of the network activations, which should prevent spikes in the OOD activations from leading the final network layer to provide high-confidence predictions for them. With a similar goal, ASH [35] filters out the majority of the activations, keeping only those representing a specific percentile.

All the methods listed till here are applied as post-hoc approaches on closed-set classifiers, which means that they do not require a specific training strategy. Other algorithms however do, for example LogitNorm [166] relies on a normalization of the logits vector during training, which should reduce the network overconfidence. Even this method can be considered part of the *discriminative* family, whose members have the significant advantage of obtaining normality predictions from a closed-set classifier, an ability that allows to use them also for Open-set recognition

Density and Reconstruction Based Methods Generative learning models, unlike discriminative ones, aim at modeling the whole distribution of known data. This should

enable identifying OOD samples in different ways. Some models try to estimate directly the likelihood of samples [180], while others exploit the sample reconstruction quality [1].

Outlier Exposure and OOD data generation Some approaches try to exploit available *outlier* data to prepare the model to recognize OOD samples [51, 179], even if the concept of what is not In-Distribution is generally quite broad. When it is not possible to access outlier samples during training, some models rely on their synthesis [25, 182].

Representation and Distance Based Methods As a good data representation should be behind any good machine learning model, some approaches focus on enhancing it to ease the detection of unknown samples. Indeed, in a reliable feature space, OOD data should lie away from known classes enabling the use of distance metrics for OOD detection. In order to reach this result, literature methods tackle the problem from two different points of view: the improvement of representation learning and the definition of specific distance metrics. Prototype learning methods, together with self-supervised ones, are examples of the first approach [152, 64]. On the other hand, researchers have proposed to use metrics like L2 distance from known samples [149], layer-wise Mahalanobis [79], or a similarity measure based on Gram matrices [142].

Chapter 3

Domain adaptation of an object detector on a single target sample

This chapter analyzes the **one-shot unsupervised cross-domain detection** research setting, designed to study those real-world scenarios in which an object detector faces a continuously varying visual distribution after deployment. This is exactly what happens in the context of social media monitoring: images uploaded on social media platforms come from a large number of parties, including individual users, and inevitably present an equally large variety of visual styles and characteristics. The most effective strategy to process a stream of images of this kind is to exploit each individual test sample as a source of information about the visual distribution from which the sample comes from, and thus to use it for model adaptation before performing predictions. We propose a solution to reach this result which exploits self-supervision to obtain adaptation on individual unlabeled samples, while we prove the unsuitability of traditional domain adaptation strategies in the setting. We also present an extension to the initial proposed solution, exploiting meta-learning to improve adaptation speed, by preparing the object detection model to the particular inference procedure that it will have to perform after deployment. A comprehensive experimental section concludes the analysis, proving the effectiveness of the proposed algorithms in the considered scenario.

Part of the work described in this chapter has been previously published in two papers:

- [33] A. D'Innocente, F. Cappio Borlino, S. Bucci, B. Caputo, and T. Tommasi
  One-shot unsupervised cross-domain detection
  European Conference on Computer Vision, ECCV 2020

- [21] F. Cappio Borlino, S. Polizzotto, B. Caputo, and T. Tommasi
  Self-Supervision & Meta-Learning for One-Shot Unsupervised Cross-Domain Detection
  Computer Vision and Image Understanding Journal (CVIU), 2022

3.1 Object detection and social media monitoring

Every day millions of images are uploaded to social media platforms by a variety of actors, from corporations to political parties, journalists and newspapers, institutions, entrepreneurs and most importantly private users. For the sake of freedom of expression, control over this content is limited, and a significant part of it is uploaded without any textual description. Still, some kind of monitoring may be necessary, first of all for tracking fake news, illegal data, and hate content. Other use cases involve different parties, like corporations, which may be interested in brand sentiment analysis, or institutions, which want to track trends of interest.

Given a continuous flow of images, an effective monitoring system has to focus not only on metadata but also on images' content, with the purpose of automatically associating them with as many relevant tags as possible. One of the possible paths to obtain this result involves training and deploying an *object detector*.

The object detection task has as its goal the recognition in images of all the objects belonging to a predefined set of categories and their localization through bounding boxes. This task has been largely investigated since the infancy of Computer Vision [160] and continues to attract a large attention in the current deep learning era [94, 134, 22]. Most of the proposed algorithms assume that training and test data come from the same visual domain; this is a strong assumption that in many deployment cases does not hold. For this reason, some researchers have started to investigate the more challenging scenario where the detector is trained on data from a source visual domain, and deployed at test time on a different target one [27, 68]. This setting is usually referred to as *cross-domain detection* and heavily relies on concepts and results from the domain adaptation literature (see Sec. 2.2.2). Unfortunately, even this approach is not suitable for social media monitoring applications. Indeed, if we consider, for instance, the scenario depicted in Figure 3.1, where there is an incoming stream of images from various social media platforms and the detector is tasked to look for instances of the class bicycle, we can immediately notice that, even if all the images contain objects belonging to this class, these instances present styles and visual appearances that are radically different. The underlying cause of this phenomenon can be found in the nature of the data source, which by definition is not uniform, but on the contrary, collects the contributions of a large variety of actors. Because each social media user expresses himself through the content that he shares, each new image may be radically different from the previously analyzed ones. In practice, in this scenario, *each image comes from a different visual domain, distinct from the visual domain where the detector has been trained.*

This poses **two key challenges** to *traditional* cross-domain detectors:

- in order to adapt to the target distribution these algorithms need, first of all, to wait and collect a sufficient amount of target data samples;
- after adaptation on target images collected up to time t, there is no guarantee that

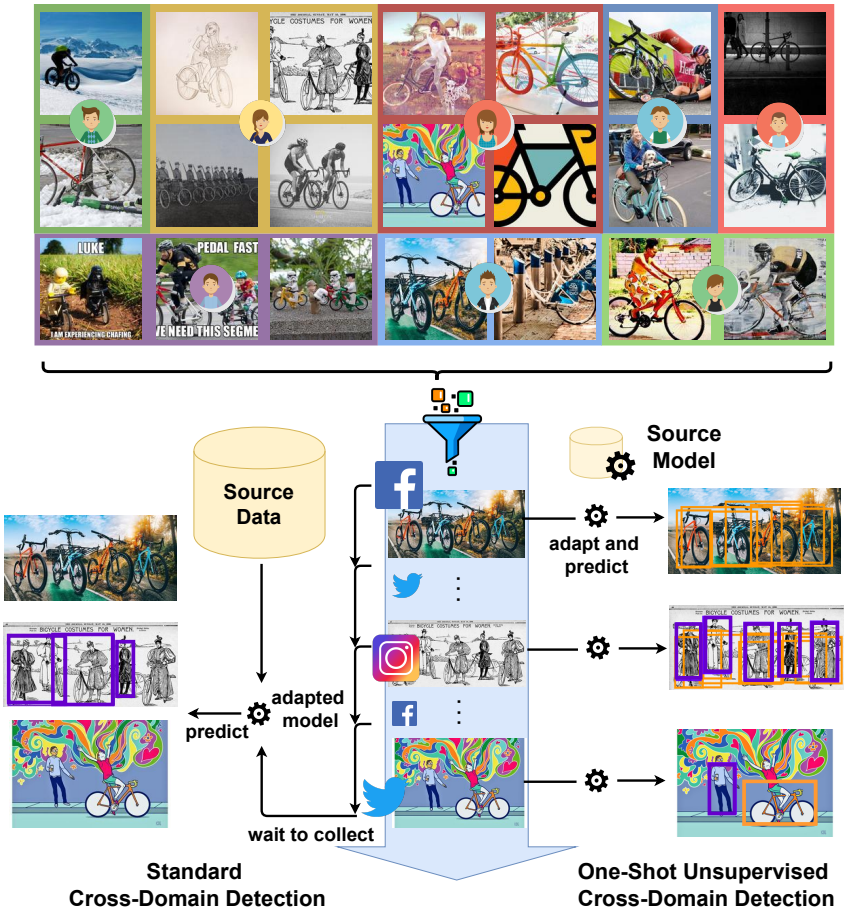


Figure 3.1: Social media images come from a variety of domains, as they are uploaded by different users. Standard Cross-Domain Detection algorithms struggle in this stream monitoring setting: they need to wait to collect sufficient data for adaptation, after which the adapted model may still be unfit to be applied on a continuously changing target domain. We propose to use each single test image for a fast adaptation of the source model, eventually providing predictions tailored for each target domain.

the images that will arrive from time $t+1$ will come from the same target domain.

We study this problem by proposing a new paradigm for dealing with it. We build on the idea that, because each target image may come from a different visual domain, the best strategy to handle them is to adapt the model to each single target image right before using it for inference.

The contributions of this chapter include:

- the definition of a new research setting, which we call **One-Shot Unsupervised Cross-Domain detection**. This is a cross-domain detection scenario where the

target domain changes from sample to sample, hence adaptation should be performed on single images;

- in order to tackle the novel problem, we propose a solution exploiting self-supervision and cross-task pseudo-labeling to enable the adaptation of an object detector on a single test sample. Our approach, which we call **OSHOT**, is trained on the labeled source domain to jointly solve two tasks at a time: the main object detection one, and an auxiliary self-supervised rotation-recognition task. After deployment, OSHOT exploits the self-supervision signal to adapt the model's backbone to the target domain of each test image, effectively improving the quality of the predictions;

- after highlighting some downsides of OSHOT, we propose an extension that we name **FULL-OSHOT**. This new version of our approach leverages a novel meta-learning formulation, designed to better combine the main supervised detection task with the self-supervised auxiliary objective. The final aim is to effectively prepare the model for the specific conditions that it will face after deployment. This improvement allows for significantly faster adaptation in the inference procedure;

- we present a novel experimental benchmark exploiting both existing databases and a **new test set** explicitly collected to study the social media monitoring scenario. On this testbed, we evaluate our approaches comparing them against recent algorithms in cross-domain adaptative detection and draw a comprehensive picture of the state-of-the-art in the novel setting.

3.1.1 Related Works

Object Detection

Modern object detectors can be divided into *one-stage* and *two-stage* techniques. In the former, classification and bounding box definition are performed on the convolution feature map either solving a regression problem on grid cells [133], or exploiting anchor boxes at different scales and aspect ratios [94]. In the latter, an initial stage deals with the region proposal process and is followed by a refinement stage that adjusts the coarse region localization and classifies the box content [134]. More recently a new transformer-based object detection paradigm has been proposed [22]. The novel structure allows discarding many hand-designed components of previous detection architectures in favor of a more streamlined pipeline in which the object detection task is framed as a set-prediction problem.

Regardless of the specific implementation, the detectors' robustness across visual domains remains a major issue.

Cross-Domain Detection

Most of the literature studying the visual domain shift problem has focused on object classification with solutions based on feature alignment or adversarial approaches, as seen in Sec. 2.2.2. More recently, the interest in cross-domain analysis has reached also the object detection world, with methods been developed considering mainly three directions:

- the inclusion of multiple and increasingly more localized feature alignment modules at different internal stages. This approach was initially proposed in [27], where Chen *et al.* pointed out, for the first time, the importance, in the object detection context, of considering both global and local domain adaptation. The Strong-Weak (SW) method[138] improves over the previous one pointing out the need for a better balancing in the alignment, with *strong global* and *weak local* adaptation.

- the adoption of a pixel-level adaptation strategy like CycleGAN [193], which enables training the detector on labeled images that look like samples coming from the target domain. This solution was initially proposed for object detection by Inoue *et al.* in [57], and extended to a full domain randomization procedure by Kim *et al.* in [68];

- the introduction of pseudo-labeling, also known as *self-training*, a procedure consisting in using the output of the source model detector as a coarse annotation to perform training on unlabeled target data. This strategy is often used in conjunction with regularization approaches in order to reduce the negative influence of noisy annotations [67, 65].

Besides these three main categories, some researchers also proposed entirely different approaches, like the ICR-CCR method presented by Xu *et al.* [171], which includes an image-level multi-label classifier and a module imposing consistency between image-level and instance-level predictions. Another strategy has been proposed by Wu *et al.* in [168] where vector decomposition is exploited to separate domain-invariant and domain-specific representations.

Adaptive Learning on a Budget

In the presence of a domain shift, learning on a target budget is extremely challenging. Indeed, as we have seen in Sec. 2.2.2, the standard assumption in UDA, is that a large amount of unsupervised target samples are available at training time so that a source model can capture the target domain style from them and adapt to it.

Only a few attempts have been made to reduce the target cardinality. In [113] the considered setting is that of *few-shot supervised domain adaptation*: only a few target samples are available but they are fully labeled. In [5, 30] the focus is on *one-shot*

unsupervised style transfer with a large source dataset and a single unsupervised target image. These works propose time-costly autoencoder-based methods to generate a version of the target image that maintains its content but visually resembles the source in its global appearance. This solution, although designed for image generation, and thus with no discriminative purpose in mind, can be used for adapting the target samples to the source domain before performing predictions on them.

Self-Supervised Learning

Unsupervised data is rich in structural information which can be uncovered by self-supervision. The usual application of this procedure is model's pre-training, which means adopting a self-supervised task for a representation learning session which is later usually followed by fine-tuning on downstream tasks. Standard self-supervised tasks used for this purpose include rotation recognition [44], and contrastive learning [26]. A number of works also indicated that self-supervision supports adaptation and generalization when combined with supervised learning in a multi-task framework [23, 13].

Meta-learning

Meta-learning or *learning to learn* is the process of training a model with the aim of preparing it for fast adaptability. One of the most important works on the topic is MAML [39]. This approach implements an *inner* learning loop designed to solve a standard supervised task, while an *outer* meta-learning loop updates the base model by observing multiple episodes of the inner task training in order to accomplish a higher level objective, with examples being a stronger generalization ability and a faster adaptation speed. Given that this technique enables adaptation on *data-scarce* tasks it has been mostly used for few-shot learning.

3.2 Problem formalization

We formalize the *one-shot unsupervised cross-domain detection* scenario:

- **data available at training time**: a set of N sample-label pairs $\{(\boldsymbol{x}_i^{(s)}, y_i^{(s)})\}_{i=1}^{N}$ with images coming from source domain S: $\boldsymbol{x}^{(s)} \in S$. Here the structured labels $y^{(s)} = (c, b)$ describe class identity c and bounding box location b for each object in image $\boldsymbol{x}^{(s)}$
- **goal**: to perform an object detection prediction on the test sample $\boldsymbol{x}^{(t)}$
- **data available at inference time**: the single test sample $\boldsymbol{x}^{(t)}$ that comes from a target domain T, with $T \neq S$;

Our base strategy to deal with this task consists of designing and training an object detection model that supports performing a short adaptation phase on the target sample before exploiting it for prediction. This strategy aims at minimizing the impact of the domain shift on the prediction for the single sample while maintaining enough flexibility to deal with the broad range of target domains that could be met after deployment.

3.3 OSHOT: adaptation on a single sample through self-supervision

In order to enable adaptation on a single target sample we propose to equip an object detector with the ability to perform an auxiliary self-supervised task besides the main detection one. We obtain this result by designing an **architecture** composed of a shared backbone and two task-specific heads. In the **pre-training** phase, we exploit the available labeled source data to train our model jointly on both tasks. At **inference** time the self-supervised head can be exploited to perform a few gradient updates on the test sample in order to **adapt** the shared backbone to the target distribution before performing a prediction. We call our approach **OSHOT**.

3.3.1 Multi-task architecture

We adopt Faster R-CNN [134] as our base detection model. It is a two-stage detector with three main components: a backbone G_f mainly composed of convolutional layers, a region proposal network (RPN), and a region-of-interest (ROI) based detection head G_d. The backbone G_f exploits its parameters θ_f to extract feature representations for input images $\boldsymbol{x}$. These feature maps are then passed to the RPN, which produces candidate object proposals according to which interesting region-level features are extracted through a procedure called ROI-pooling. These ROI-features are processed by the detection head G_d. We use θ_d to indicate the parameters of RPN and ROI. At training time the RPN and detection head outputs are used to compute the detection loss $\mathscr{L}_d$. This training objective combines the loss of both RPN and ROI, each of them composed of two terms:

$$\begin{aligned}\mathscr{L}_d(G_d(G_f(\boldsymbol{x}|\theta_f)|\theta_d), y) = &\big(\mathscr{L}_{class}(c^*) + \mathscr{L}_{regr}(b)\big)_{RPN} + \\ &\big(\mathscr{L}_{class}(c) + \mathscr{L}_{regr}(b)\big)_{ROI}\,.\end{aligned} \tag{3.1}$$

Where $\mathscr{L}_{class}$ is a classification loss to evaluate the object recognition accuracy, while $\mathscr{L}_{regr}$ is a regression loss on the box coordinates that enables refining the localization. We use c^* to highlight that RPN deals with a **binary** classification task to separate foreground and background objects, while ROI deals with the multi-class objective needed to discriminate among c foreground object categories.

We **extend the standard Faster R-CNN architecture** by adding a second head after the initial backbone. This head is designed to perform the self-supervised rotation recognition task [44]. We indicate it as G_r and its parameters as θ_r.

3.3.2 Multi-task pretraining

The purpose of this phase is to build an object detection model, trained on source data, but ready to be adapted on a single target sample after deployment. Our multi-task network is trained jointly on the main detection task and on the auxiliary self-supervised one. Formally, to each source training image $\boldsymbol{x}^{(s)}$ we apply four geometric transformations $R(\boldsymbol{x}, \alpha)$ where $\alpha = q \times 90°$ indicates rotations with $q \in \{1, \dots, 4\}$.

In this way, we obtain a new set of sample-label pairs $\{(R(\boldsymbol{x})_j, q_j)\}_{j=1}^{M}$, where we dropped the α without loss of generality. Here q_j is the rotation label, it encodes which transformation was applied to the original image in order to obtain the rotated version $R(\boldsymbol{x})_j$. For the multi-task training, we adopt this learning objective:

$$\begin{aligned} \underset{\theta_f, \theta_d, \theta_r}{\operatorname{argmin}} \sum_{i=1}^{N} \mathscr{L}_d(G_d(G_f(\boldsymbol{x}_i^{(s)}|\theta_f)|\theta_d), y_i^{(s)}) + \\ \lambda \sum_{j=1}^{M} \mathscr{L}_r(G_r(G_f(R(\boldsymbol{x}^{(s)})_j|\theta_f)|\theta_r), q_j) \end{aligned} \tag{3.2}$$

where $\mathscr{L}_r$ is the cross-entropy loss, used to train the model on the rotation recognition task, cast as a classification problem with four classes. In order to keep a balance between rotated and non-rotated samples in the multi-task training we randomly pick only one rotation angle per instance.

When designing the rotation recognition part of our network we consider two different approaches:

- the naïf approach consists in applying the rotation recognition head G_r directly on the backbone output. In this case, G_r attends to the whole image and can thus use also background information (e.g. the position of the sky) to recognize the rotation applied;
- a more advanced approach, which is also more consistent with the processing strategy used by the detection head, consists of applying ROI-pooling to extract the features corresponding to a specific bounding box as a preliminary step and then passing these features to G_r. In this case, the rotation recognition head attends to single-object regions and avoids using noisy information from the background to solve its task.

In OSHOT we adopt the second approach (*i.e.* box rotation) but consider also the first as part of our ablation analysis whose results are in Section 3.5.

3.3.3 Test-time adaptation and inference

At inference time, before performing predictions on each target sample $\boldsymbol{x}^{(t)}$, we fine-tune the backbone's parameters θ_f by performing some gradient update steps on the

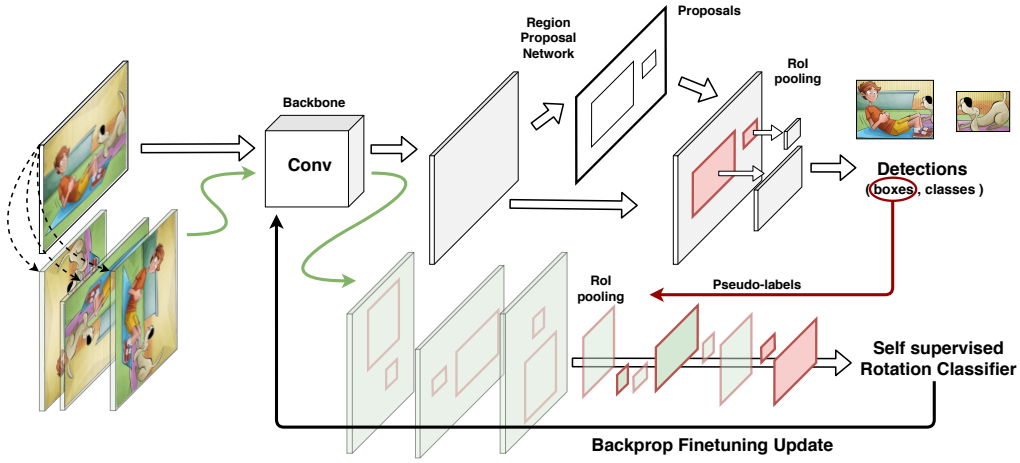


Figure 3.2: Visualization of the adaptive phase of OSHOT with cross-task pseudo-labeling. The target image passes through the network and produces detections. While the class information is not used, the identified boxes are exploited to select object regions from the feature maps of the rotated image. The obtained box-specific feature vectors are finally sent to the rotation classifier. A number of subsequent fine-tuning iterations allows to adapt the convolutional backbone to the domain represented by the test image

self-supervised task. This enables adapting the original feature representation to the visual distribution of the new sample. Specifically, we start from the rotated versions $R(\boldsymbol{x}^{(t)})$ of the provided sample and optimize the rotation classifier through:

$$\underset{\theta_f,\theta_r}{\operatorname{argmin}} \mathscr{L}_r(G_r(G_f(R(\boldsymbol{x}^{(t)})|\theta_f)|\theta_r), q^{(t)}) \tag{3.3}$$

This process involves only G_f and G_r, while the RPN and ROI detection components, described by G_d, remain unchanged. In this adaptation phase, we include dropout to prevent overfitting on the single target sample. In the following we use γ to indicate the number of gradient steps (*i.e.* iterations), with $\gamma = 0$ corresponding to the output of the OSHOT pre-training phase (*i.e.* the source model).

At the end of the fine-tuning process, the inner feature model is described by θ_f^* and the detection prediction on $\boldsymbol{x}^{(t)}$ can be performed by computing

$$\hat{y}^{(t)} = G_d(G_f(\boldsymbol{x}^{(t)}|\theta_f^*)|\theta_d) \,. \tag{3.4}$$

Cross-task pseudo-labeling As in the pre-training phase, also in the adaptation stage, we have two possible choices to design G_r: either considering the whole image features or focusing on the object locations. In order to implement the second approach, since in this case we are not provided with ground truth information about

Algorithm 1: Adaptive phase on one target sample

Input: G_f, G_d, G_r, parameters θ_f, θ_d, θ_r, from the pre-training phase, rotator R
Data: Target image $\boldsymbol{x}^{(t)}$
$(\theta_f^*, \theta_r^*) \leftarrow (\theta_f, \theta_r)$ ▷ `copy params`
while *still* γ *iterations* **do**
  $\tilde{c}, \tilde{b} \leftarrow G_d(G_f(\boldsymbol{x}^{(t)}|\theta_f^*)|\theta_d)$
  $\boldsymbol{x}_r^{(t)} \leftarrow R(\boldsymbol{x}^{(t)})$; $b_r \leftarrow R(\tilde{b})$ ▷ `rand. rotation` q
  minimize self-supervised loss using b_r for ROI-pooling:
  $(\theta_f^*, \theta_r^*) \leftarrow (\theta_f^*, \theta_r^*) - \alpha \nabla_{\theta_f^*, \theta_r^*} \mathscr{L}_r(G_r(G_f(\boldsymbol{x}_r^{(t)}|\theta_f^*)|\theta_r^*), q)$
end
final detection prediction using updated parameters $\hat{y}^{(t)} = G_d(G_f(\boldsymbol{x}^{(t)}|\theta_f^*)|\theta_d)$

the position of the objects inside the target image, we adopt a particular form of *cross-task self-training*. Specifically, we follow the self-training strategy used in [67, 57] with a cross-task variant: instead of reusing the pseudo-labels produced by the source model on the target to update the detector, we exploit them for the self-supervised rotation classifier. This means that in the adaptation phase, we use the prediction output of the detection head to select locations to be used for object-level rotation recognition. In this way, we keep the advantage of the self-training strategy, but because we do not use predicted class labels as pseudo-labels we largely reduce the risks of error propagation due to wrong predictions.

More practically, starting from the model parametrized by (θ_f, θ_d) we obtain the feature maps from all the rotated versions of the target sample $G_f(\{R(\boldsymbol{x}^{(t)}), q\}|\theta_f)$, with $q = 1, \dots, 4$. The feature map produced by the original image (*i.e.* $q = 4$) is provided as input to the RPN and ROI network components to get the predicted detection $\tilde{y}^{(t)} = (\tilde{c}, \tilde{b}) = G_d(G_f(\boldsymbol{x}^{(t)}|\theta_f)|\theta_d)$. This pseudo-label is composed of the class label $\tilde{c}$ and the bounding box location $\tilde{b}$. We discard the first and consider only the second to localize the region containing an object in all four feature maps, also recalibrating the position to compensate for the orientation of each map. This information is then processed through ROI-pooling and the obtained box-specific features are passed to the rotation classification head G_r. This process is graphically represented in Figure 3.2, while the whole test-time adaptation and subsequent inference pipeline is summarized in Algorithm 1.

3.4 FULL-OSHOT: meta-learning prepares adaptation

In the *one-shot unsupervised cross-domain detection* setting, models are allowed to adapt to each target sample visual domain before performing a prediction. Still, this process should be as fast as possible to enable the deployment of such models for real-world monitoring applications. By design, OSHOT is able to adapt to a single target sample by exploiting its self-supervised branch. Still, adaptation may require up to $\gamma = 30$ iterations, in order for the results to improve significantly. At the same time, OSHOT presents an asymmetry between the multi-task pre-training and the adaptation phase: in the former, the self-supervised task and the main detection one are learned *jointly*, in the latter the self-supervised task is used alone for adaptation, and detection is performed only as a second step.

In order to overcome this asymmetry, and, at the same time, obtain a model which is able to adapt with fewer iterations, we propose to extend OSHOT, by introducing after the original multi-task pre-training an additional pre-training phase based on meta-learning. In this phase, we exploit source data to prepare the model for the test time adaptation procedure. In particular, by using a single training sample at a time but transformed via data augmentation, we simulate cross-domain learning episodes in which the self-supervised task is employed in an inner optimization loop that performs features adaptation, while an outer loop optimizes the network parameters in order to obtain the **best detection performance** on top of the adapted features. We call **FULL-OSHOT** this variant of our method.

3.4.1 Meta-learning pre-training

With the goal of preparing our model for the particular test-time adaptation procedure that it has to face at inference time, we introduce a second pre-training phase built on top of the bi-level optimization process of MAML [39]. Specifically, we propose to meta-train the detection model with the rotation task as its inner base learner. The optimization objective can be written as

$$\begin{aligned} &\underset{\theta_f,\theta_d}{\operatorname{argmin}} \frac{1}{K}\sum_{k=1}^{K} \mathscr{L}_d(G_d(G_f(\boldsymbol{x}_k|\theta'_f)|\theta_d), y) \\ \text{s.t. } (\theta'_f, \theta'_r) = &\underset{\theta_f,\theta_r}{\operatorname{argmin}} \mathscr{L}_r(G_r(G_f(R(\boldsymbol{x}_k)|\theta_f)|\theta_r), q) \end{aligned} \tag{3.5}$$

In words, we start by focusing on the rotation recognition task for each source sample $\boldsymbol{x}$ after augmenting it in $k = 1, \dots, K$ different ways. We consider semantic-preserving augmentations (*e.g.* gray-scale, color jittering) and perform multiple learning iterations (η gradient-based update steps). This optimization, whose learning objective is reported in the second row of Equation (3.5), leads to the update of the feature extractor and rotation classification modules (parameters θ'_f and θ'_r). The outer meta-learning

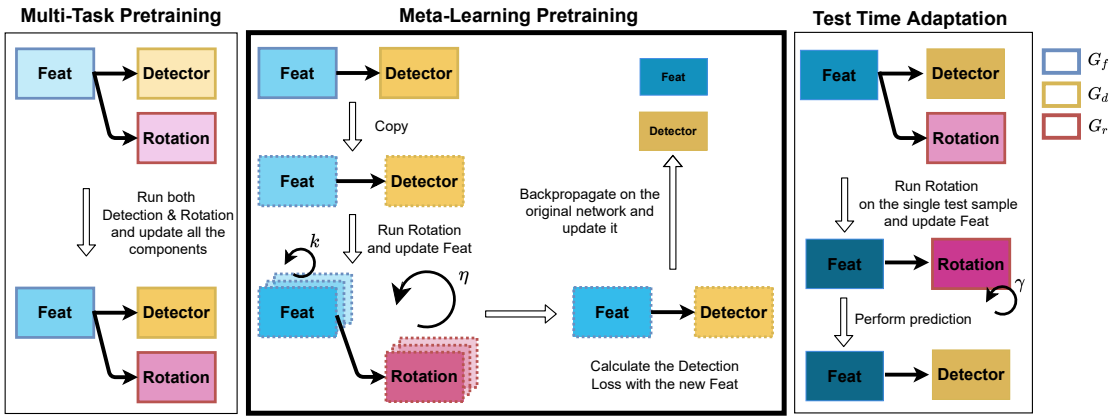


Figure 3.3: Visualization of the complete FULL-OSHOT proposed approach. The first phase, inherited from OSHOT, is a multi-task pre-training in which all the modules are updated through the rotation and detection losses. The second pre-training stage, introduced for FULL-OSHOT, exploits a meta-learning with the rotation recognition as the inner optimization task to prepare the network for the test-time adaptation stage. In this last phase, both the rotation and feature extractor modules are updated by performing the self-supervised task iteratively on a single test sample. The adapted feature extractor is finally used to predict on the same test image. Each change in color shades indicates an update of a module. We use dotted lines to highlight the components optimized in the meta-learning loop.

loop, whose learning objective is in the first row of the same Equation, leverages the adapted features to optimize the detection model over all the K data variants. In order to simulate the deployment setting we neglect the ground truth object location for the inner rotation objective. Instead, to get object locations we adopt the same cross-task pseudo-labeling procedure that is used in the test-time adaptation phase.

We integrate this meta-learning pre-training phase in the source model preparation strategy, after the multi-task pre-training introduced in OSHOT. We summarize in Fig. 3.3 the complete FULL-OSHOT pipeline. Given that, in the meta-learning pre-training, we have introduced two novel components, *i.e.* the meta-learning loop and the additional augmentations, it is important to assess if the improvements guaranteed by this pre-training phase come from only one of the components, or from both. Thus, with the goal of conducting a comprehensive analysis, we also consider two intermediate models: *Tran*-OSHOT extends OSHOT with the data semantic-preserving transformations used in FULL-OSHOT, and *Meta*-OSHOT corresponds to FULL-OSHOT without transformations (*i.e.* $K = 1$).

The meta-learning strategy is summarized in Algorithm 2, the adaptation phase is inherited as is from OSHOT (see Algorithm 1).

Algorithm 2: Meta-learning on one source sample

Input: G_f, G_d, G_r, parameters θ_f, θ_d, θ_r, rotator R, augmenter A
Data: Source image $\boldsymbol{x}$ with $y = (b, c)$

while *still k augmentations* **do**
  $\boldsymbol{x}_k \leftarrow A(\boldsymbol{x})$
  $(\theta'_f, \theta'_r) \leftarrow (\theta_f, \theta_r)$ ▷ `copy params`
  while *still η iterations* **do**
    $\tilde{c}, \tilde{b} \leftarrow G_d(G_f(\boldsymbol{x}_k|\theta'_f)|\theta_d)$
    $\boldsymbol{x}_{r,k} \leftarrow R(\boldsymbol{x}_k)$; $b_{r,k} \leftarrow R(\tilde{b})$ ▷ `rand. rotation` q
    minimize self-supervised loss using $b_{r,k}$ for ROI-pooling:
    $(\theta'_f, \theta'_r) \leftarrow (\theta'_f, \theta'_r) - \alpha \nabla_{\theta'_f, \theta'_r} \mathscr{L}_r(G_r(G_f(\boldsymbol{x}_{r,k}|\theta'_f)|\theta'_r), q)$
  end
  compute the supervised loss $l_k = \mathscr{L}_d(G_d(G_f(\boldsymbol{x}_k|\theta'_f)|\theta_d), y)$
end
minimize the supervised loss $(\theta^*_f, \theta^*_d) \leftarrow (\theta_f, \theta_d) - \beta \nabla_{\theta_f, \theta_d} \sum_{k \in K} l_k$

3.5 Experimental results

3.5.1 Datasets

We run an extensive experimental analysis considering several datasets.

Real-world (Pascal-VOC) Pascal-VOC [38] is one of the standard real-world image datasets for object detection benchmarks. Its two versions *VOC2007* and *VOC2012* both contain bounding boxes annotations of 20 common categories. VOC2007 has 5011 images in the train-val split and 4952 images in the test split, while VOC2012 contains 11540 images in the train-val split.

Artistic Media Datasets (AMD) Clipart1k, Comic2k, and Watercolor2k [57] are three object detection datasets designed for benchmarking domain-adaptive detection methods when the source domain is Pascal-VOC. Clipart1k shares with it its 20 categories and it is composed of 500 images in the training set and 500 images in the test set. Comic2k and Watercolor2k both have the same 6 classes (a subset of the 20 classes of Pascal-VOC), and 1000-1000 samples in the training-test splits each.

Cityscapes [31] It is an urban street scene dataset with pixel level annotations of 8 categories. It has 2975 and 500 images respectively in the training and validation splits. We use the instance level pixel annotations to generate bounding boxes of objects.

Foggy Cityscapes [140] It is obtained by adding different levels of synthetic fog to Cityscapes images. We only consider images with the highest amount of artificial fog, thus training-validation splits have 2975-500 images respectively.

Social Bikes In order to accurately evaluate the performance of algorithms in the novel one-shot unsupervised cross-domain detection setting, we collect a dedicated dataset containing 530 images of scenes with persons/bicycles collected from Twitter (now *X*), Instagram, and Facebook by searching for *#bike* tags. The top part of Figure 3.1 shows some examples extracted from this collection. We designed this dataset to be used as a target when the source domain is Pascal-VOC, indeed the two classes it contains, *i.e. person* and *bicycle*, are shared with that dataset. With respect to the other testbeds, Social Bikes covers a larger variety of visual styles related to the tastes and preferences of each social media user.

3.5.2 Baselines and competitors

In order to build a comprehensive picture of the state-of-the-art, we compare OSHOT and its variants with a number of different methods. For all the algorithms considered

we use the same ResNet-50 [48] backbone pre-trained on ImageNet1k [34] with the goal of performing fair comparisons.

Baselines Our main *Baseline* is Faster-RCNN trained on the source domain and deployed on the target without further adaptation. We also build an alternative *Tran-Baseline*, a variant obtained by applying at training time, over the original baseline, the same data semantic-preserving transformations introduced in FULL-OSHOT. Its purpose is to assess how much improvement is obtained due to data augmentation rather than due to the training strategy.

Competitors Besides the baselines, we consider a set of literature methods. *Div-Match* [68] is a cross-domain detection algorithm that, by exploiting target data, creates multiple randomized domains via CycleGAN and aligns their representations using an adversarial loss. *SW* (StrongWeak) [138] aligns source and target features by balancing the weight of global and local adaptation. *SW-ICR-CCR* [171] adds on top of *SW* an image-level multi-label classifier and a module imposing consistency between the image-level and instance-level predictions. *ICCR-VDD* [168] uses a vector-decomposition technique to disentangle domain-invariant and domain-specific features. This enables using only relevant features to extract object proposals in a cross-domain setting.

3.5.3 Implementation details

For all the experiments the standard transformation pipeline includes resizing the shorter image's side to 600p and performing random horizontal flipping. When doing multi-task pre-training we set the weight λ to 0.05. Our model is robust to the exact value of this parameter in $[0.01, 0.2]$: the relevance of the rotation recognition objective should be high enough for the auxiliary task to be learned, but low enough to not hijack the main task learning.

The multi-task pre-training phase of **OSHOT** is carried out for 70k iterations using SGD with momentum set at 0.9, the initial learning rate is 0.001, and decays by a factor of 10 after 50k iterations. We use a batch size of 1, keep batch normalization layers fixed for both pre-training and adaptation phases, and freeze the first 2 blocks of ResNet50, as is standard practice in the field.

For what concerns **FULL-OSHOT** there are two pre-training steps. For the first 60k iterations, the training is identical to that of OSHOT, while in the last 10k iterations, the meta-learning procedure is activated. The inner loop optimization on the self-supervised task runs with $\eta = 5$ iterations and the batch size is 2 to accommodate for two transformations of the original image. Specifically, we used gray-scale and color-jitter with brightness, contrast, saturation, and hue all set to 0.4. All the other hyperparameters remain unchanged w.r.t. OSHOT.

***Tran*-OSHOT** differs from OSHOT only for the last 10k learning iterations, where the batch size is 2 and the network sees images augmented using the same transformations of FULL-OSHOT.

***Meta*-OSHOT** is instead identical to FULL-OSHOT, with the only exception that transformations are dropped, thus the batch size is 1 also in the last 10k pre-training iterations.

3.5.4 Evaluation protocol

The performance evaluation is carried out mainly by reporting the mean Average Precision (**mAP**, see definition in Sec. A.2.2), with results averaged over three runs. We also conduct a detailed error analysis using TIDE [8]. This is a toolbox designed to estimate how much each type of detection failure contributes to the missing mAP. The main reason for using TIDE is that it provides visualizations giving not only qualitative but also quantitative insights, by counting False Positives, False Negatives, and identifying six categories of errors:

- **classification error** (*Cls*): for objects localized correctly ($IoU \geq 0.5$) but classified incorrectly;
- **localization error** (*Loc*): for objects classified correctly but localized incorrectly ($0.1 \leq IoU < 0.5$);
- **classification and localization error** (*Both*): for objects mislocalized and misclassified at the same time;
- **duplicate** (*Dupe*): for correct detection of objects whose ground truth bounding box has already been associated with another higher scoring detection;
- **background** (*Bkg*): for detection of background as foreground ($IoU < 0.1$);
- **missed** (*Miss*): for all the undetected ground truth boxes not covered by other types of errors.

We also include some qualitative detection results in Figure 3.4.

Settings definition We compare the methods described before by evaluating their performance in three main settings: **i)** adapting to social feeds, **ii)** adapting to large distribution shifts, and **iii)** adapting to adverse weather. These are all cross-domain settings, thus we use the notation *Source* $\rightarrow$ *Target* to identify them. Because the state-of-the-art cross-domain detection algorithms that we use as reference were not designed to manage adaptation on a single unlabeled target image and may fail in this condition, we test them by following slightly different protocols w.r.t. OSHOT and its variants:

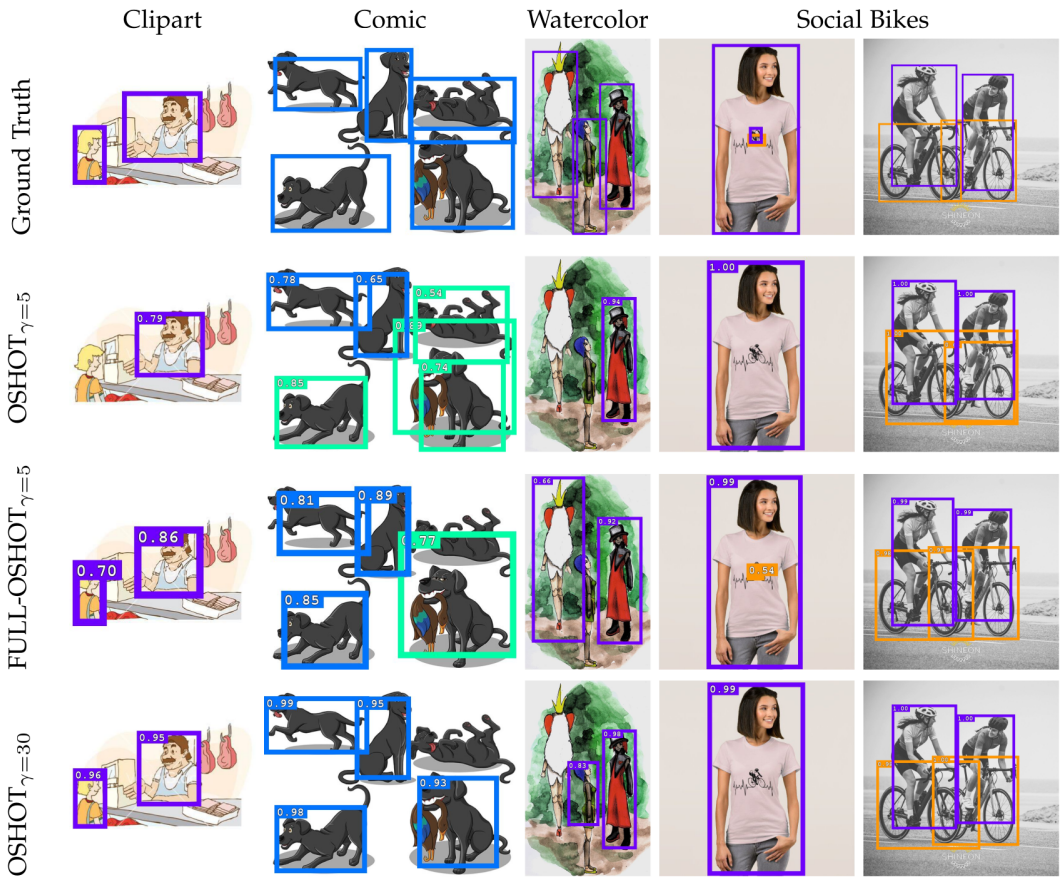


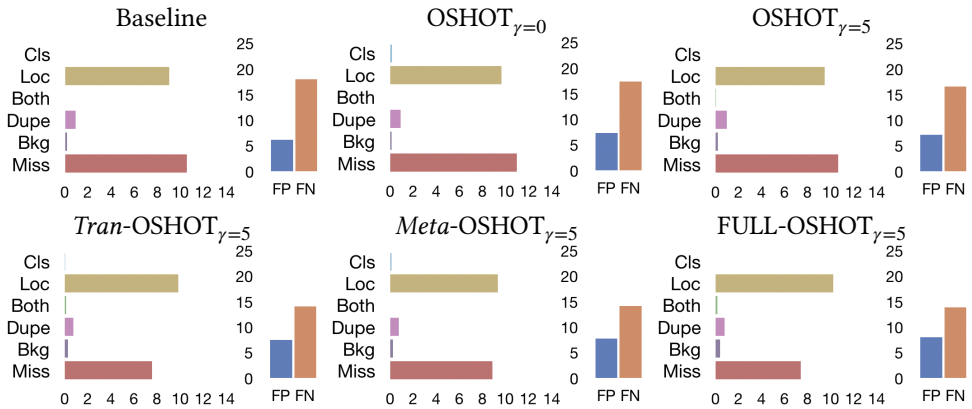
Figure 3.4: Some detection results examples of FULL-OSHOT and OSHOT when changing the number of adaptive iterations.

- for OSHOT and its variants, the training is carried out considering access to **source data only**. At inference time a single test sample at a time is considered, the source model is adapted to it and then a prediction is performed before considering the next sample;
- cross-domain detection algorithms need to access **target data at training time**, so they operate from an advantaged position, having access to ten randomly selected target samples (in the *Ten-Shot Target* scenario) or even to the entire target set (in the *Whole Target* scenario) during training. We collect average precision statistics during inference.

Table 3.1: Results for VOC → Social Bikes

Method		person	bicycle	mAP
One-Shot *Target*				
Baseline		69.0	74.1	71.6
Tran-Baseline		71.4	74.2	72.8
$\gamma = 0$	OSHOT	68.9	74.6	71.8
	Tran-OSHOT	71.6	74.0	72.8
	Meta-OSHOT	69.5	73.5	71.5
	FULL-OSHOT	71.7	74.3	73.0
$\gamma = 5$	OSHOT	72.1	74.9	73.5
	Tran-OSHOT	73.0	74.7	73.9
	Meta-OSHOT	72.6	74.5	73.6
	FULL-OSHOT	73.3	75.1	74.2
Ten-Shot *Target*				
DivMatch [68]		69.5	73.1	71.3
SW [138]		69.4	73.0	71.2
SW-ICR-CCR [171]		72.5	77.6	75.1
VDD-DAOD [168]		68.8	75.3	72.1
Whole *Target*				
DivMatch [68]		73.6	77.1	75.4
SW [138]		68.6	70.3	69.5
SW-ICR-CCR [171]		72.0	72.8	72.4
ICCR-VDD [168]		71.1	71.9	71.5

Table 3.2: TIDE-based [8] detection error analysis for VOC → Social Bikes



3.5.5 Adapting to social feeds

When the data comes from multiple providers, the assumption that all target images originate from the same underlying distribution does not hold and standard cross-domain detection methods are penalized regardless of the number of seen target samples. To analyze the performance in this setting we use Pascal-VOC as source domain and Social Bikes as target.

In Table 3.1 the mAP results with $\gamma = 0$ allow us to compare the pre-training models before adaptation and already show the advantage of FULL-OSHOT over OSHOT, as well as over the *Tran* and *Meta* variants. When $\gamma = 5$ all variants of OSHOT obtain an improvement that ranges from 1.9 (OSHOT) to 2.6 (FULL-OSHOT) points over the Baseline just by adapting on a single test sample for a small number of iterations. Despite granting them access to the whole set of adaptation samples, the reference domain adaptive algorithms reach at best an advantage of 1.2 points over FULL-OSHOT. When using ten target samples, half of the methods show a *negative transfer* with respect to w.r.t. the Baseline.

By looking at the detection error analysis in Table 3.2 we can see that the adaptation iterations allow OSHOT to reduce the number of false negatives. Moreover, both *Tran*-OSHOT and FULL-OSHOT obtain a higher mAP than OSHOT thanks to lower *Miss* errors. The performance of FULL-OSHOT confirms that the meta-learning strategy with semantic-preserving data augmentations successfully prepares the model for the test-time adaptation procedure.

3.5.6 Adapting to large distribution shifts

Artistic images represent a difficult testbed for cross-domain methods, as they show perturbations in shape and color which are challenging for detectors trained only on real-world photos. We investigate this setting by training on Pascal-VOC and testing on the Artistic Media Datasets (AMD): Clipart, Comic, and Watercolor. The results in Table 3.3 show that OSHOT and its variants, by only exploiting one sample at a time, and few adaptive iterations ($\gamma = 5$), outperform the adaptive detectors which leverage on ten target samples. It is interesting to notice that none of the adaptive detectors is able to work in data-scarce conditions: indeed, they all obtain results comparable to those of the *Tran*-Baseline and of the pre-training phase of our approach ($\gamma = 0$). We also highlight that when $\gamma = 5$, *Meta*-OSHOT obtains results higher than *Tran*-OSHOT and only slightly lower on average than FULL-OSHOT. This proves that the meta-learning strategy alone (without additional data augmentation) effectively prepares the detector for test-time adaptation.

From the detection error analysis in Table 3.4, we see that the data augmentation of *Tran*-OSHOT pushes for a lower number of errors of type *Miss*, while the meta-learning strategy of *Meta*-OSHOT gets a lower number of *Classification* errors. FULL-OSHOT takes advantage of both, obtaining the best performance.

Table 3.3: Results for VOC → AMD

	(a) VOC → Clipart		(b) VOC → Comic	(c) VOC → Watercolor
	One-Shot *Target*		***One-Shot*** *Target*	***One-Shot*** *Target*
	Method	mAP	mAP	mAP
	Baseline	26.4	18.1	42.8
	Tran-Baseline	27.6	22.4	46.3
$\gamma = 0$	OSHOT	28.8	19.9	45.7
	Tran-OSHOT	28.6	20.1	45.4
	Meta-OSHOT	29.4	20.2	45.8
	FULL-OSHOT	28.6	21.1	46.4
$\gamma = 5$	OSHOT	30.8	22.3	48.1
	Tran-OSHOT	30.5	24.9	47.7
	Meta-OSHOT	31.4	24.8	49.0
	FULL-OSHOT	31.7	25.2	48.9
	Ten-Shot *Target*		***Ten-Shot*** *Target*	***Ten-Shot*** *Target*
	DivMatch [68]	26.3	20.8	45.4
	SW [138]	26.4	21.0	42.0
	SW-ICR-CCR [171]	27.2	21.1	45.3
	ICCR-VDD [168]	27.6	24.8	43.1

3.5.7 Adverse weather

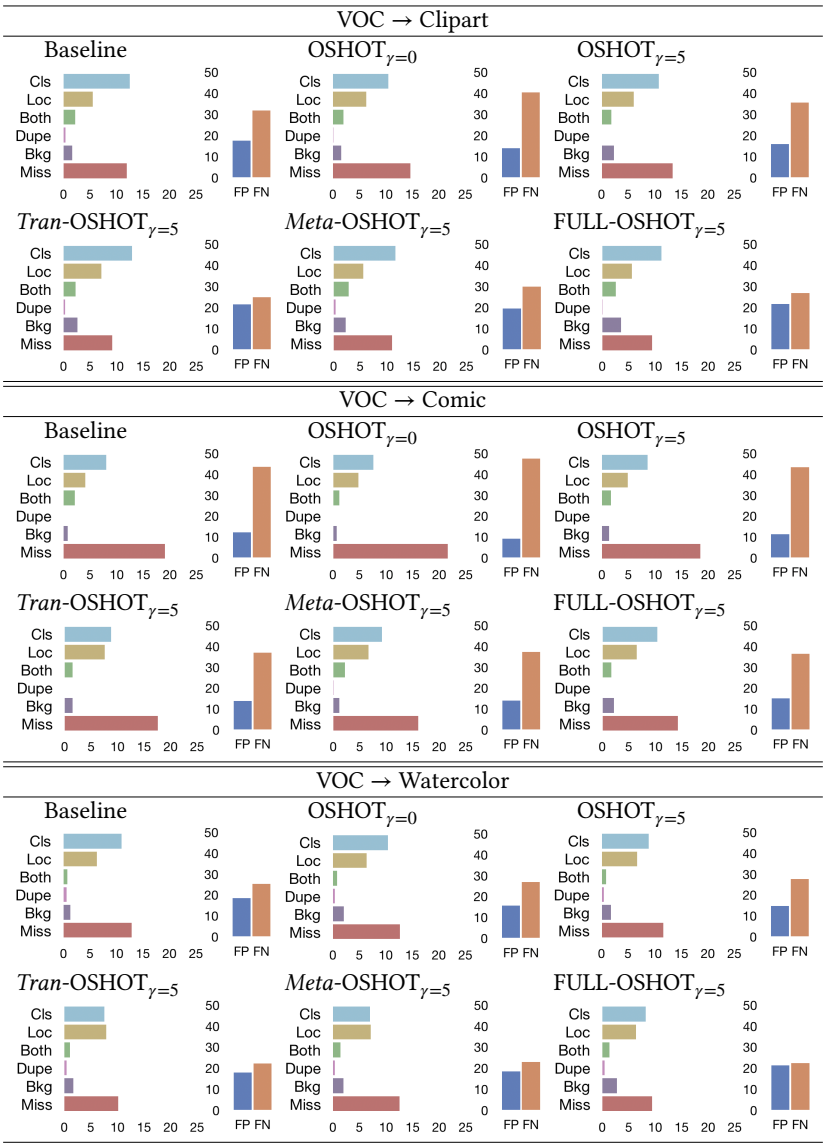
Some environmental conditions, such as the presence of fog, may be disregarded in source data acquisition, yet generalization to these circumstances is crucial in real-world applications. We test this scenario by considering the Cityscapes → FoggyCityscapes setting. We perform model selection on the Cityscapes validation split before deployment.

The results in Table 3.5 show that domain adaptive detectors struggle in this scenario. Only SW-ICR-CCR and VDD-DAOD are able to exploit the small adaptation set and obtain a meaningful improvement over the Baseline. For what concerns OSHOT and its variants, the pretraining alone ($\gamma = 0$) helps in gaining a better generalization ability: all variants but *Meta*-OSHOT show higher performance than the Baseline. By looking at the error analysis in Table 3.6 we notice an improvement also in the *Miss* error type which decreases when passing from the Baseline to OSHOT $\gamma = 0$, reaching its lower value for FULL-OSHOT with $\gamma = 5$.

3.6 Analysis

We perform some additional analyses in order to provide a more complete picture of the performance of OSHOT and its variants, and of their inner workings.

Table 3.4: TIDE-based [8] detection error analysis for VOC → AMD



3.6.1 Comparison with One-Shot Style Transfer

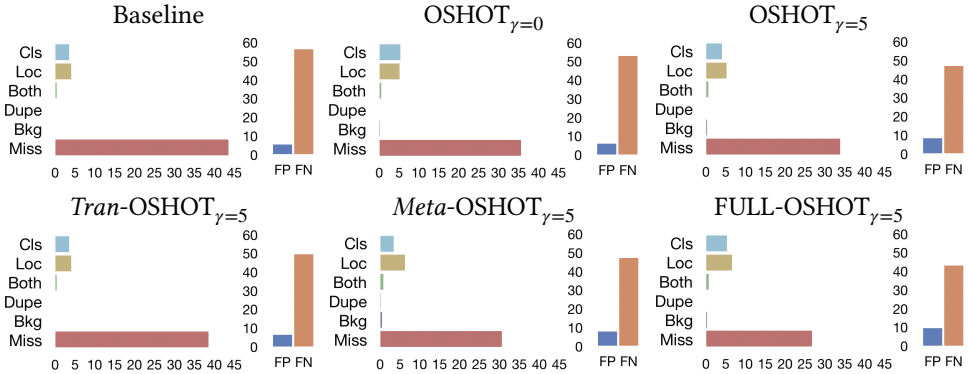
Although they are not designed for the one-shot cross-domain scenario, it is possible to apply one-shot style transfer methods as an alternative solution in our setting. We experiment with BiOST [30], using it to modify the style of the target sample towards that of the source domain before performing inference. Due to the time-heavy requirements to run this method on each test sample, we test it only on Social Bikes and on a random subset of 100 Clipart images that we name Clipart100. We compare the performance and time requirements of our approach w.r.t. BiOST on these two targets.

Table 3.7 shows that on Clipart100 the Baseline obtains 27.9 mAP and BiOST has

Table 3.5: Results for Cityscapes → FoggyCityscapes

	Method	person	rider	car	truck	bus	train	mcycle	bicycle	mAP
***One-Shot** Target*										
	Baseline	30.4	36.3	41.4	18.5	32.8	9.1	20.3	25.9	26.8
	Tran-Baseline	32.1	35.2	42.9	17.8	31.0	4.3	22.6	30.0	27.0
$\gamma = 0$	OSHOT	32.2	38.6	39.0	20.5	30.6	12.9	22.4	31.2	28.4
	Tran-OSHOT	30.5	37.4	42.7	16.9	29.5	14.5	21.9	30.4	28.0
	Meta-OSHOT	30.6	35.1	35.9	16.6	28.4	7.6	18.2	28.4	25.1
	FULL-OSHOT	31.7	40.8	43.7	18.3	28.8	11.0	22.8	33.3	28.8
$\gamma = 5$	OSHOT	32.7	39.3	41.1	21.1	33.1	12.6	22.7	31.9	29.3
	Tran-OSHOT	30.9	38.5	43.0	17.5	32.1	13.9	21.6	30.5	28.5
	Meta-OSHOT	32.1	38.2	39.9	17.4	30.9	7.5	21.0	29.2	27.0
	FULL-OSHOT	32.0	39.7	43.8	18.8	31.8	10.6	22.1	33.2	29.0
***Ten-Shot** Target*										
	DivMatch [68]	27.6	38.1	42.9	17.1	27.6	14.3	14.6	32.8	26.9
	SW [138]	25.5	30.8	40.4	21.1	26.1	34.5	6.1	13.4	24.7
	SW-ICR-CCR [171]	29.6	40.8	39.6	20.5	32.8	11.1	24.0	34.0	29.1
	ICCR-VDD [168]	32.3	32.1	41.7	25.0	29.0	40.0	12.6	19.7	29.0

Table 3.6: TIDE-based [8] detection error analysis for Cityscapes → FoggyCityscapes



an advantage over it of 1.9 points. Conversely, on the Social Bikes dataset, BiOST incurs a slight negative transfer, which highlights its inability to effectively modify the source's style on this more challenging testbed. OSHOT improves over the baseline on Clipart100 but its mAP remains lower than that of BiOST, while it outperforms both the baseline and BiOST on Social Bikes. Finally, FULL-OSHOT shows the best results on both the datasets. The last row of the table presents the time complexity of all the considered methods, which is identical for OSHOT and FULL-OSHOT since the number of adaptive iterations is the same. BiOST instead, needs more than six hours to modify the style of a single source instance. Moreover, we highlight that BiOST works under the

Table 3.7: Comparison between baseline, one-shot style transfer and our approach in the one-shot unsupervised cross-domain detection setting. Speed computed on an RTX2080Ti with full precision settings

	Baseline	BiOST [30]	OSHOT $\gamma = 5$	FULL-OSHOT $\gamma = 5$
mAP on Clipart100	27.9	29.8	28.2	30.4
mAP on Social Bikes	71.6	71.4	73.5	74.2
Adaptation time (s per sample)	-	2.4×10^4	1.3	1.3

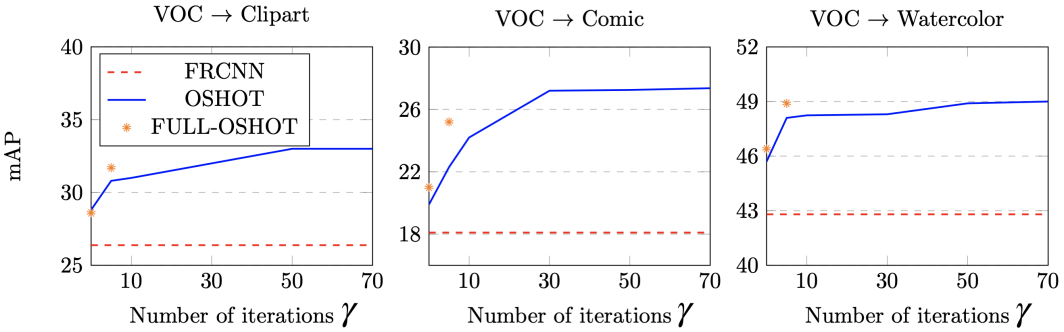


Figure 3.5: Performance of OSHOT at different number of adaptive iterations.

strict assumption of accessing at the same time the entire source training set and the target sample, while OSHOT and its variants do not need to access the source dataset in the adaptation phase.

3.6.2 Increasing the number of Adaptive Iterations

The bi-level optimization process at the basis of meta-learning requires non-trivial computational and memory burdens that might limit the feasible number of iterations η. In FULL-OSHOT we use the same conditions for the meta-learning pre-training and test-time adaptation phases, thus we set a small number of training steps with $\gamma = \eta$. This choice, however, does not limit the effectiveness of the method, which becomes clear when comparing it with OSHOT at an increasing number of iterations. We studied the mAP performance of OSHOT on the AMD dataset and collected the results in Figure 3.5. We observe a positive correlation between the number of fine-tuning iterations and the mAP of the model in the earliest steps, while the performance generally reaches a plateau after about 30 iterations: increasing γ beyond this value does not affect significantly the final results. From the plots, we can see that the performance of FULL-OSHOT with just 5 adaptation iterations can be achieved and eventually surpassed by the standard OSHOT only at the cost of a much higher number of adaptation iterations. This behavior is also reflected by the visualizations in Figure 3.4 where the

Table 3.8: Rotating image vs rotating objects on OSHOT

	$G_r(image)$	$G_r(box)$
VOC → Clipart	31.0	**33.9**
VOC → Comic	21.0	**26.9**
VOC → Watercolor	48.2	**52.0**
Cityscapes → Foggy Cityscapes	27.7	**31.9**

results obtained by FULL-OSHOT with $\gamma = 5$ are more similar to those obtained by OSHOT with $\gamma = 30$ than those obtained by OSHOT with $\gamma = 5$. These results highlight that the meta-learning approach, at the cost of a higher train-time computational cost, is able to produce a model that effectively adapts to the target sample domain with just a few iterations, leading to an inference-time computational cost reduction and speed improvement.

3.6.3 Rotation recognition localization

As described in Section 3.3 there are two main options when designing the strategy to integrate the rotation recognition self-supervised task in an object detection pipeline: either considering the whole image, or focusing on the object-level, by selecting box-features obtained by applying ROI-pooling using the objects' bounding boxes (ground truth ones in the pre-training phase, and predicted ones as part of the cross-task pseudo-labeling strategy in the adaptive phase).

We adopted the second strategy (box-rotation) in all the experiments presented till here as we argue that, by solving the auxiliary task on object-level features, we prevent the network from using background features for predictions, which may be misleading or provide shortcuts for performing the tasks without learning anything useful (*e.g.* : finding fixed patterns in images, exploiting watermarks). We validate our choice by comparing it against using the rotation task on the entire image in both training and adaptation phases. Table 3.8 shows results for Pascal-VOC → AMD and Cityscapes → Foggy Cityscapes using OSHOT with $\gamma = 30$. We observe that the choice of box-rotation is critical for the effectiveness of the algorithm. Indeed, adopting this strategy rather than whole-image rotation results in mAP improvements that range from 2.9 to 5.9 points, indicating that this approach leads to learning better features for the main task.

In Figure 3.6 we visualize Grad-CAM [143] heatmaps for a set of samples from our datasets, in order to see which image regions are attended by the network when performing the rotation recognition task. By comparing the column on the left, which shows results obtained by a network trained to perform rotation recognition on the whole image, with the column on the right, which shows the results obtained by a network trained using the box-rotation strategy, it is easy to notice some differences in the behavior. In particular, the adoption of the box-rotation strategy leads the network to

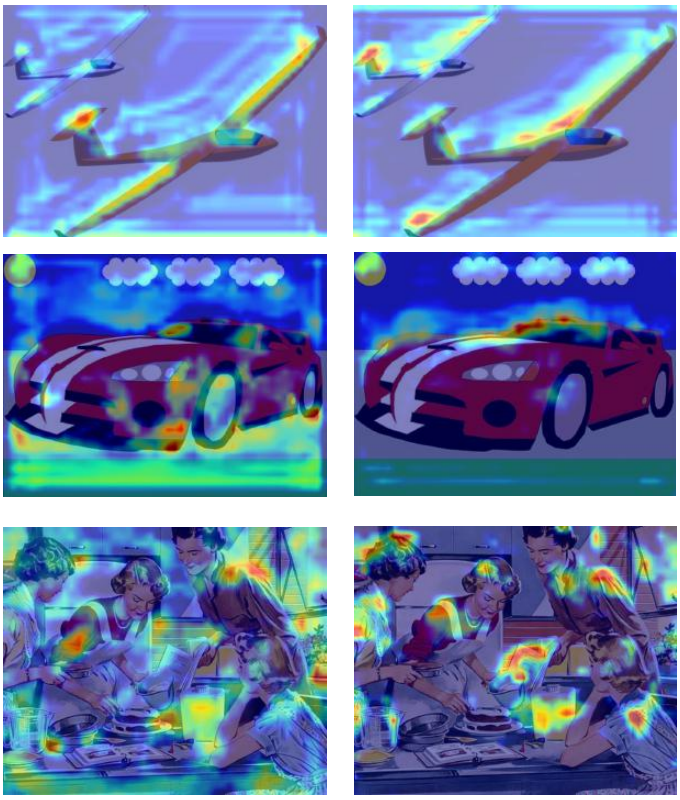


Figure 3.6: Grad-CAM [143] heatmaps highlight the parts of the images which are mostly used by the network to perform rotation recognition. Comparison between a network trained with $G_r(image)$ (left), and one trained with $G_r(box)$ (right).

focus more on local objects' features, while the network trained with image-rotation employs a more global focus attending to various parts of the image and thus also focusing on background features.

3.7 Conclusions

This chapter has focused on the proposal and study of a new cross-domain analysis research setting called *one-shot unsupervised cross-domain detection*. This novel research problem is designed to study those real-world scenarios in which the deployment conditions involve a non-homogeneous and continuously varying target visual domain, for which traditional domain-adaptive object detection strategies may fail. An example of this situation is the visual social media monitoring task, where a stream of user-uploaded images has to be analyzed taking into account that every sample could come from its own visual domain.

The chapter proposed a solution for the novel task built on top of a multi-task joint self-supervised and supervised learning strategy. This approach enables a source-trained model to be adapted on a single test sample at a time, right before the inference step, in order to obtain predictions tailored for any target domain the model may face. An improvement over this first solution has also been proposed, exploiting a meta-learning-based pre-training phase designed explicitly to prepare the model for the test-time adaptation procedure which is used at inference time. Thanks to this preparation the model is able to improve the adaptation speed and the final performance.

The two proposed approaches are deeply analyzed through a comprehensive experimental evaluation which has not only proved their effectiveness in dealing with the novel *one-shot unsupervised cross-domain detection task* but also their superiority w.r.t. state-of-the-art domain-adaptive methods, which have shown to be extremely brittle when they need to adapt in data-scarce scenarios.

These results highlight that the assumption of having access to a large set of target samples at training time does not always hold, which points out the need to design specific solutions to deal with this situation. In this context, OSHOT [33] has been one of the first approaches proposed to perform adaptation at test time on a single target sample, together with a concurrent work proposing a test time training procedure [150]. In the last years the research in this field has been particularly active, sparking the need for the presentation of a survey on the topic [89].

Chapter 4

Data augmentation and Domain Generalization: an unbiased perspective

In this chapter, we focus on **Domain Generalization**, in particular **analyzing its relationship with data augmentation**. The latter is a standard practice applied during the training of neural networks in order to reduce overfitting and regularize learning, and it is particularly effective, especially in those cases in which the training dataset is quite small. However, the impact that special data augmentation transformations could have on the domain-shift robustness of the learned features is often neglected, and the DG literature prefers focusing on the development of strategies that obtain domain invariance through more sophisticated approaches. We perform here a thorough analysis of the situation, by first proposing a very simple style-transfer-based data augmentation pipeline that enables obtaining a robust DG baseline outperforming previous state-of-the-art approaches, and then experimenting with combinations of this augmentation pipeline with DG algorithms that tackle the problem with seemingly *orthogonal* strategies. The main outcome of our analysis is that many state-of-the-art methods are not able to provide any advantage once they are applied on top of the improved baseline, a phenomenon that should push for the development of novel strategies able to reach this result, eventually providing even more robust cross-domain performances.

Part of the work described in this chapter has been previously published in:
[19] F. Cappio Borlino, A. D'Innocente, and T. Tommasi
Rethinking Domain Generalization Baselines
25th International Conference on Pattern Recognition, ICPR 2020

4.1 Data augmentation enables generalization

Domain generalization is one of the most important among the research settings that fall under the umbrella of cross-domain analysis, with the main reason being the fact that it covers a wide number of real-world scenarios. Indeed, **in many deployment conditions, the target visual domain is different from the source one**, which makes it fundamental to adopt a deep model showing great robustness to visual domain changes. Domain generalization is however a challenging task, as the target data is fed to the system only after deployment. In order to build robust models a common solution relies on the exploitation of multiple available sources during training, a strategy that enables understanding which are the domain-invariant features that are really relevant for the task, in the **hypothesis that analogous invariances will hold for future test domains**. Towards this goal, most of the existing DG strategies try to incorporate the observed data invariances, capturing them at the feature-level [84] or at the model one, using meta-learning [82] or self-supervision [23, 172].

An alternative solution consists in **extending the source domains by synthesizing new images**, with the aim of better spanning the data space and including a larger variability in the training set. This is usually done by learning generative models with the specific constraint of preserving the image content while varying its global appearance. With the recent improvements in generative learning, the adoption of this kind of approach for DG is becoming more and more viable, with results that seem to be particularly effective [189]. However, the performance of this kind of solution tends to grow together with the **complexity** of the employed learning procedure, which in many cases may involve one or multiple generator modules and adversarial training protocols.

The radical difference between the data augmentation-based strategies and the feature- and model-based ones has initiated a particular trend in the most recent domain generalization research. Several papers focusing on data augmentation solutions discuss their merit in comparison with feature and model-based techniques [189, 185]. Still, newly introduced feature and model-based approaches avoid benchmarks against data augmentation strategies, probably considering them unfair competitors due to the extended training set [56].

This trend has led to **a split in the literature**, which is a source of confusion about the real state-of-the-art. With the analysis that we carry out in this chapter, we aim at recomposing the field, by clarifying which are the potentialities of the data augmentation strategy and which should be its relationship with model- and feature-based techniques.

The main contributions of this chapter are:

- the proposal of **a simple and effective style-transfer-based data augmentation approach for domain generalization**. This method uses AdaIN [55], a model for real-time style-transfer, by re-purposing it for data augmentation with

the aim of combining semantic and texture information of the available sources (see Figure 4.1)

- the design of tailored strategies to integrate style-transfer-based data augmentation with current state-of-the-art approaches;
- a comprehensive analysis of the DG performance of the novel approach of style-transfer-based data augmentation both when used alone and in combination with state-of-the-art methods. This analysis points out that the original advantage of those methods almost always disappears when compared with the data-augmented baseline.

The scenario described by this analysis clearly suggests the need for rethinking domain generalization baselines. On one side, simple data augmentation strategies should be envisaged to increase source data variability compatible with orthogonal feature and model generalization approaches. On the other, new adaptive strategies should be designed to build over images generated by style-transfer approaches.

4.1.1 Background and problem formalization

In this chapter, we focus on object recognition performed in the multi-source domain generalization setting. For a general background presentation and a formal description of the problem we refer to Sec. 2.2.1. As presented there, the most adopted DG approaches rely on complex training procedures, designed specifically to improve generalization by focusing on the feature-level, for example through an explicit domain-invariance enforcement, or the model-level, when adopting self-supervised training procedures or meta-learning based ones.

Besides these approaches, there is an alternative research line that focuses on a potentially **orthogonal paradigm**, *i.e.* **extending the training dataset through data augmentation** techniques designed specifically to increase the source diversity: a model learned on the augmented samples gains robustness against the specific features that they show. Several approaches have been proposed to generate new samples, from the simple random changing of color or background in the case of synthetic objects and robotics applications [153], to the more complex use of adversarial training [162, 189]. Domain Mixup can also be included in the data augmentation methods [174]: pairs of examples from different domains are interpolated together with their label to learn on a more continuous domain-invariant data distribution. In general, any approach for pixel-level alignment can be used for data augmentation. Some of these are complex GAN-based approaches that learn to replicate the visual appearance of a specific domain or set of domains [193], while others focus on transferring the style of a single image to another one, by using features statistics as a style summary, as proposed in AdaIN [55]. This last method is not only flexible in terms of support of a wide variety of visual

Figure 4.1: Source augmentation by style-transfer allows us to generate different variants of each image, borrowing the style from any other image and by keeping the original semantic content. In this example the images come from the OfficeHome dataset and the style-transfer is performed using AdaIN.

styles, but it is also able to perform style-transfer in a single forward pass, enabling real-time performance and its integration in an online data augmentation pipeline.

4.2 Source augmentation by style-transfer

Let's consider a basic deep learning classifier $C_{\theta_c}(\cdot)$, parametrized by θ_c, and trained on the source data in a supervised fashion:

$$\underset{\theta_c}{\operatorname{argmin}} \sum_{i=1}^{n_S} \sum_{j=1}^{N_{S_i}} \mathscr{L}\left(C_{\theta_c}\left(\boldsymbol{x}_j^{(s)}\right), y_j^{(s)}\right)$$

With the objective of increasing data variability, we study how to augment each sample $\boldsymbol{x}^{(s)}$ by keeping its semantic content and changing the image style, borrowing it from the other available source data. The plethora of stylized samples $\tilde{\boldsymbol{x}}_k^{(s)}$ (with k identifying a style image) obtained from $\boldsymbol{x}^{(s)}$ inherit the original label $y^{(s)}$ and enrich the training set, possibly making the model learned by optimizing on the original and augmented data more robust to domain shifts. Thus, our analysis will consider a two-step process, where a deep model A_{θ_a} parametrized by θ_a is first learned on the source data to perform style transfer $\boldsymbol{x}^{(s)} \rightarrow \tilde{\boldsymbol{x}}^{(s)} = A_{\theta_a}(\boldsymbol{x}^{(s)})$, and then it is used to perform data augmentation online while C_{θ_c} learns to classify the image content.

4.2.1 Style Transfer Model

We adopt AdaIN [55] as a stylization method. It is a simple and effective encoder-decoder-based approach that supports performing style-transfer in real-time, by transferring the visual style of a *style image* onto a *content image*, in a content-preserving fashion. The idea at the basis of AdaIN is that the style of an image is encoded in the statistics (mean and standard deviation) of its features.

Stylization strategy and architecture The encoder E extracts representative features from the content image c and the style image s:

$$f_c = E(c)\,; \qquad f_s = E(s)\,;$$

Content features f_c are then re-normalized to have the same channel-wise mean and standard deviation of the style features f_s:

$$f_{cs} = \sigma(f_s)\left(\frac{f_c - \mu(f_c)}{\sigma(f_c)}\right) + \mu(f_s)\,. \tag{4.1}$$

where $\mu(\cdot)$ is the mean and $\sigma(\cdot)$ the standard deviation.

Finally, the obtained features f_{cs} are mapped back to the image space through the decoder $\boldsymbol{D}$.

Model training The overall learning objective exploited by AdaIN is composed of two losses:

$$\mathscr{L}_A = \mathscr{L}_c + \lambda \mathscr{L}_s. \tag{4.2}$$

The content loss $\mathscr{L}_c$ is the euclidean distance between the stylization step output f_{cs} and a new encoding of the decoded output:

$$\mathscr{L}_c = \|E(D(f_{cs})) - f_{cs}\|_2 \tag{4.3}$$

The style loss $\mathscr{L}_s$ adopts a similar logic, but it is computed by measuring the difference in terms of mean and standard deviation of the ReLU output of several encoder layers $\{\phi_i\}_{i=1}^{L}$:

$$\mathscr{L}_s = \sum_{i=1}^{L} \|\mu(\phi_i(D(f_{cs}))) - \mu(\phi_i(s))\|_2 + \sum_{i=1}^{L} \|\sigma(\phi_i(D(f_{cs}))) - \sigma(\phi_i(s))\|_2 \tag{4.4}$$

In words, this loss pushes the network to produce features statistics as close as possible for the original image and the stylized one.

The method has two main hyperparameters $\{\lambda, \alpha\}$. The first controls the weight of the style loss during training and is generally kept fixed at $\lambda = 10$. The second enables a test-time control of a content-style trade-off by interpolating between the feature maps that are fed to the decoder with

$$f_{cs\alpha} = D((1-\alpha)f_c + \alpha f_{cs}) \tag{4.5}$$

4.2.2 Style Transfer as Data Augmentation

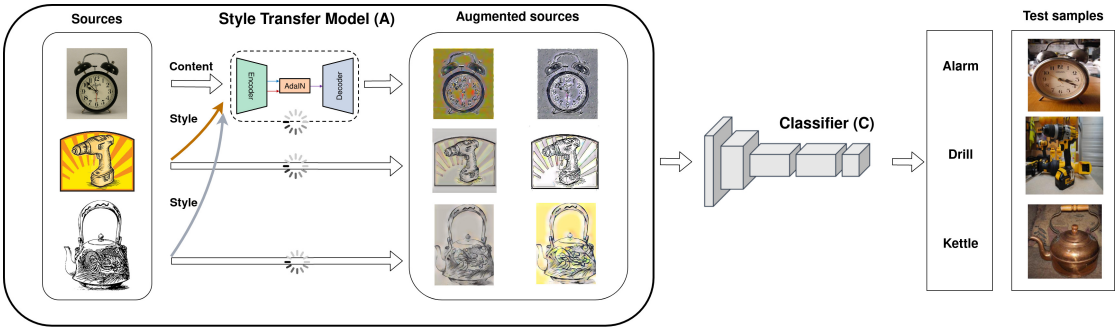


Figure 4.2: The classifier's training pipeline. Each training sample is augmented by borrowing the style from another image of the batch.

When training our object classifier C_{θ_c} we use data batches containing samples extracted from all the source domains. Before performing the network forward, the samples of the batch are augmented as depicted in Figure 4.2. In practice, each sample has

the role of content image for the stylization procedure, and any of the remaining instances in the same batch can be selected randomly to work as a style provider. In this scenario, stylization can happen both from images of the same source domain (*e.g.* two photos) or from images of different domains (*e.g.* a photo and a painting). To regulate this process we use a stochastic approach with the transformed image $\tilde{\boldsymbol{x}}^{(s)}$ replacing its original version $\boldsymbol{x}^{(s)}$ with probability p.

4.3 Experimental results

The purpose of this experimental analysis is to run a thorough evaluation of the impact of style-transfer-based data augmentation on domain generalization. We are mainly interested in observing how this data augmentation can improve the standard baseline model, how it compares with the most recent state-of-the-art DG methods, and how their combination performs. In the following we provide details on the chosen benchmark and state-of-the-art models, describing how the data augmentation strategy can be integrated into each approach.

4.3.1 Datasets

We consider three standard benchmark datasets that differ in terms of number of classes and covered domains.

PACS [81] This dataset contains images of 7 object classes spanning 4 visual domains: Photo, Art painting, Cartoon, and Sketch. Given that the visual domains go from real-world representations to artistic images, the style variability is quite large. We follow the experimental protocol proposed in the original paper by training on the train splits of three source domains (using the validation splits for model selection), and then testing on the whole left-out domain which acts as unknown target.

OfficeHome [159] It is similar to PACS, it covers 4 domains (Art, Clipart, Product, and Real-World) but shows a much larger set of 65 object classes. In the experiments, we use a random 90-10 train-val split to select the training images for the 3 source domains (once again, the validation images are used for model selection) and testing is performed on the whole left-out target domain.

VLCS [154] It is built upon 4 different datasets: Pascal-VOC 2007, Labelme, Caltech, and SUN. It contains 5 object categories. Differently from the other considered testbeds, all the domains are composed of real-world photos with the shift mainly due to camera type, illumination conditions, point of view, *etc.* Moreover, while Caltech is composed of object-centered images, the other three datasets contain scene images. We apply the same experimental protocol of [23]: the predefined full training data is randomly partitioned in train and validation sets with a 90-10 ratio. The training is performed on the train splits of the 3 source domains while the validation splits are used for model selection. In the end, the model is tested on the predefined test split of the left-out domain. This split has been defined randomly by selecting 30% of images of the overall dataset.

All our results are obtained by performing the average over 3 runs. In the case of both OfficeHome and VLCS, the random 90-10 train-val split was repeated for each run.

4.3.2 Comparison methods

The main *Baseline* of our analysis is the standard approach usually adopted in DG: a classification model trained on all the available source data together, a strategy often called *Deep All* [23]. We indicate with *Original* the standard data augmentation procedure, which includes horizontal flippling and random cropping, while we use *Stylized* for the cases where we add style-transfer-based data augmentation. We evaluate under both these augmentation settings the behavior of four among the most recent DG methods. We dedicate particular attention to how we integrate the novel data augmentation strategy with each of the considered approaches as we want to get the most out of them without undermining their nature. In particular, considering that the style-transfer leads to domain mixing, we avoid integrating it in procedures that need a separation among source domains.

DG-MMLD [106] This algorithm is based on clustering and domain adversarial feature alignment. Since this method does not use the source domain labels, the integration of the proposed style-transfer-based data augmentation is straightforward and follows the strategy used for the *Baseline*: styles of random images are applied to content images (inside a batch) with probability p.

Epi-FCR [82] It is a meta-learning-based method that splits the network into two modules, each one trained by pairing it with a partner that is badly tuned for the domain considered in the current learning episode. These two modules are the *feature extractor* and the *classification* head, which alternatively cover the two roles of *learning part* and *bad reference*. In a second phase, a final model is learned by integrating the trained modules together with a random classifier used as *regularizer*. As in the first phase knowing the source domain labels is crucial to choosing and setting up the two network modules, mixing the domains with style-transfer could degrade the model performance. In the second stage, instead, all the data sources are considered together: we choose this phase for the application of the style-transfer-based data augmentation procedure.

DDAIG [189] This approach belongs to the family of data augmentation strategies, and it employs a transformation network that is trained to produce synthesized samples that keep the same label of the original image but fool a domain classifier. In the adversarial learning procedure the transformation module, the label classifier, and the domain classifier are iteratively updated. In particular, the label classifier is trained on all the source samples, both original and synthetic. We can thus further extend this set by introducing style-transfer augmented data.

Rotation The original paper [172] is one of the research publications that has shown that self-supervised knowledge supports domain generalization when combined with

supervised learning in a multi-task model. We consider a self-supervised rotation recognition task, similar to the one employed by OSHOT (see 3.3), where the orientation angle of each image should be recognized among $\{0°, 90°, 180°, 270°\}$. The model minimizes a linear combination of the supervised and self-supervised losses, with this second contribution which is assigned a weight η generally kept lower than 1 in order to let the supervised model guide the learning process. In this case, the domain labels are not used during training, so the application of the source augmentation by style-transfer is straightforward, as for the *Baseline*.

Besides the methods described till here, which are used to evaluate how style-transfer-based data augmentation can improve the performance of DG methods, we consider a further reference which is an alternative strategy for data augmentation by samples-mixing. In particular, we focus on *Mixup* [181] an approach originally defined to improve generalization in standard in-domain learning: it interpolates samples and their labels, regularizing a neural network so that it does not provide drastically changing predictions when evaluated on samples just outside the training distribution. The purpose of the training strategy is to favor a simple linear behavior in the model predictions between training examples. Its hyper-parameter $\gamma \in \{0, \infty\}$ controls the strength of interpolation between data pairs, going back to the Baseline for $\gamma = 0$. Mixup has already been used for cross-domain analysis [174], thus we follow the proposed path testing both data mixing at pixel- and feature-level.

4.3.3 Training setup

Style-transfer model A We train this model on source data before training the classification one. We adopt for it the original backbone used for AdaIN [55] which is a VGG. We train the model for 20 epochs with a learning rate equal to $5e-5$. The hyper-parameters α and p used in each experiment are specified in the caption of the respective result tables. We also perform an in-depth analysis of the sensitivity of our strategy to these parameters in Section 4.3.5.

Classification model C For this model, we use AlexNet and ResNet18 backbones. For the training of *Baseline*, *Rotation*, and *Mixup* we use SGD with 0.9 momentum for $30k$ iterations. The batch size is set to 32 images per source domain, which means a total of 96 training samples for each batch as all the testbeds considered have three source domains. The learning rate and the weight decay are respectively fixed at 0.001 and 0.0001. Regarding the hyperparameters of the individual algorithms, we empirically set the *Rotation* auxiliary weight to $\eta = 0.5$ and for *Mixup* we set $\gamma = 0.4$.

We implement *Rotation* by adding a rotation recognition branch to our *Baseline* architecture. For all the other competitors (*DG-MMLD*, *Epi-FCR*, and *DDAIG*) we use the authors' provided code and train by following the originally proposed protocols for both the *Original* and *Stylized* versions. We simply integrate the original code with the new style-transfer procedure and add different datasets/backbones where needed. We

Table 4.1: PACS classification accuracy (%). We used AdaIN with $\alpha = 1.0$ and $p = 0.75$ for AlexNet-based experiments and AdaIN with $\alpha = 1.0$ and $p = 0.90$ for those based on ResNet18.

		Painting	Cartoon	Sketch	Photo	Average
AlexNet						
Original	Baseline	66.83	70.85	59.75	89.78	71.80
	Rotation	65.66	71.89	62.15	89.88	72.39
	DG-MMLD	69.27	72.83	66.44	88.98	74.38
	Epi-FCR	64.70	72.30	65.00	86.10	72.03
	DDAIG*	62.77	67.06	58.90	86.82	68.89
Stylized	Baseline	71.96	72.47	76.47	88.34	**77.31**
	Rotation	71.74	73.39	75.98	89.22	77.59
	DG-MMLD	70.50	70.84	75.39	88.43	76.29
	Epi-FCR	65.19	69.54	71.97	83.43	72.53
	DDAIG	69.35	71.10	70.99	87.70	74.79
Mixup	pixel-level	66.03	68.00	51.18	88.90	68.53
	feature-level	67.04	69.10	55.40	88.88	70.11
ResNet18						
Original	Baseline	77.28	73.89	67.01	95.83	78.50
	Rotation	78.16	76.64	72.20	95.57	80.64
	DG-MMLD	81.28	77.16	72.29	96.06	81.83
	Epi-FCR	82.10	77.00	73.00	93.90	81.50
	DDAIG*	79.41	74.81	69.29	95.22	79.68
Stylized	Baseline	82.73	77.97	81.61	94.95	**84.32**
	Rotation	79.51	79.93	82.01	93.55	83.75
	DG-MMLD	80.85	77.10	77.69	95.11	82.69
	Epi-FCR	80.68	78.87	76.57	92.50	82.15
	DDAIG	81.02	78.75	79.67	95.07	83.63
Mixup	pixel-level	78.09	71.08	66.58	93.85	77.40
	feature-level	81.20	76.41	69.67	96.31	80.90

report the previously published results whenever possible, while we indicate with a star (*) the results that we obtained by running the authors' original code.

4.3.4 Numerical results

PACS results Table 4.1 shows the results obtained on the PACS benchmark with both AlexNet and ResNet18. These results allow us to make two considerations:

- the style-transfer-based data augmentation produces an improvement of more than 5 percentage points in the *Baseline* performance. Looking at the results for the different domains we can see that improvement is higher for Art Painting, Cartoon, and Sketch, than for Photo;

Table 4.2: OfficeHome classification accuracy (%). We used AdaIN with parameters $\alpha = 1.0$ and $p = 0.1$.

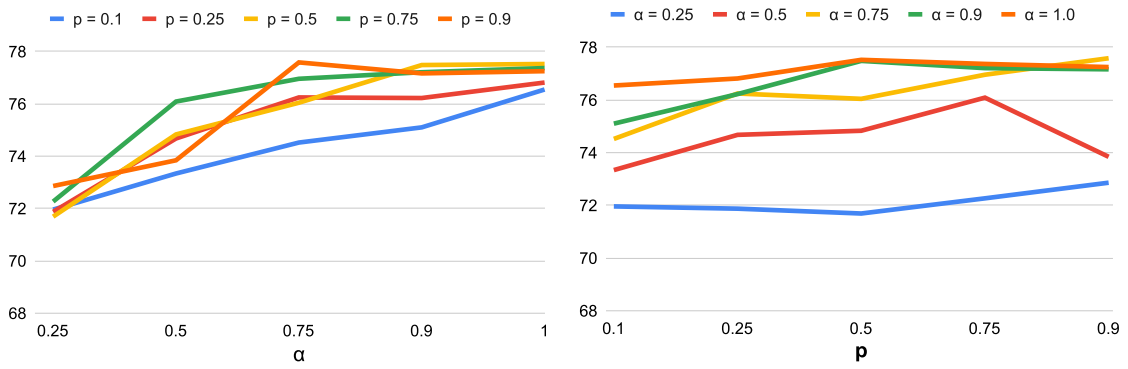
ResNet18						
		Art	Clipart	Product	Real World	Average
Original	Baseline	57.14	46.96	73.50	75.72	63.33
	Rotation	55.94	47.26	72.38	74.84	62.61
	DG-MMLD*	58.08	49.32	72.91	74.69	63.75
	Epi-FCR*	53.34	49.66	68.56	70.14	60.43
	DDAIG*	57.79	48.32	73.28	74.99	63.59
Stylized	Baseline	58.71	52.33	72.95	75.00	**64.75**
	Rotation	57.24	52.15	72.33	73.66	63.85
	DG-MMLD	59.24	49.30	73.56	75.85	64.49
	Epi-FCR	52.97	50.14	67.03	70.66	60.20
	DDAIG	58.21	50.26	73.81	74.99	64.32
Mixup	feature-level	58.33	39.76	70.96	72.07	60.28

Table 4.3: VLCS classification accuracy (%). We used AdaIN with parameters are $\alpha = 1.0$ and $p = 0.75$.

AlexNet						
		CALTECH	LABELME	PASCAL	SUN	Average
Original	Baseline	94.89	59.14	71.31	64.64	72.49
	Rotation	94.50	61.27	68.94	63.28	72.00
	DG-MMLD*	96.94	59.10	68.48	62.06	71.64
	Epi-FCR*	91.43	61.36	63.44	60.07	69.07
	DDAIG*	95.75	60.18	65.48	60.78	70.55
Stylized	Baseline	96.86	60.77	68.18	63.42	72.31
	Rotation	96.86	60.77	68.18	63.42	72.31
	DG-MMLD	97.49	61.02	64.23	62.37	71.28
	Epi-FCR	92.69	58.18	62.59	57.87	67.83
	DDAIG	97.48	60.48	65.19	62.57	71.43
Mixup	feature-level	94.73	62.15	69.82	62.98	72.42

- all the considered state-of-the-art DG methods seem to benefit from the source augmentation as in absolute terms their performance grows. However, they significantly lose in effectiveness in comparison with the baseline, as in their *Stylized* versions they cannot outperform it anymore.

OfficeHome results Table 4.2 shows the results obtained on the OfficeHome dataset with ResNet18 backbone. The numbers more or less confirm the observations done for the PACS case, with the only difference that, in this case, the improvement produced by the source augmentation by style-transfer is more limited. The *Stylized Baseline* obtains again the best accuracy outperforming the competitors, even when those are improved using the same source augmentation strategy.



(a) Influence of α

(b) Influence of p

Figure 4.3: Average accuracy on PACS with AlexNet backbone when varying the style-transfer model training's hyperparameters

VLCS results Table 4.3 reports results obtained on the VLCS benchmark with AlexNet backbone. This dataset is particularly challenging and shows a fundamental limit of tackling DG through style-transfer data augmentation. Since in this benchmark the domain shift is not due to style differences, source augmentation by style-transfer does not support generalization.

Mixup for DG Focusing on Mixup we see that the results over all the considered datasets show that it does not support generalization across domains and it might perform even worse than the *Original Baseline*. Between the two considered pixel and feature variants, only the second shows some advantage on PACS, so we focused on it for the other testbeds. Still, the results remain lower than those obtained by the DG methods both with and without style-transfer-based data augmentation.

4.3.5 Additional analyses

AdaIN hyperparameters In Figure 4.3 we see how the PACS AlexNet performance is influenced by changes in either α or p while keeping the other fixed. With a low value of α the style-transfer is too weak to produce an effective appearance change and introduce extra variability. In general, the best results are obtained using $\alpha = 1$ regardless of the specific value of p. For what concerns the latter, we can see that, if α is high enough, even a small p allows to obtain good performance, with the best results obtained with $p = 0.5$ or $p = 0.75$.

Style-transfer from external data vs source data The strategy that we propose for the applications of AdaIN differs from what appeared in previous works. Indeed, both the original approach [55] and its use for data augmentation in [185], exploit a style-transfer model that has been trained using MS-COCO [92] as the source of content

Figure 4.4: Example of application of style-transfer using AdaIN. The top left image comes from the PACS Photo domain and is used as content while the top center image comes from PACS Art Painting domain and is used as style image. On top right there is the translation performed using AdaIN trained on MS-COCO and WikiArt images. In the second row we see the translations performed using our AdaIN models trained on source data only, respectively when the Art Paintings, Cartoon, Sketch and Photo domains are left-out during model's training.

Table 4.4: Comparison of AdaIN training strategies

	Art Painting	Cartoon	Sketch	Photo	Average
Stylized Baseline	71.96	72.47	76.47	88.34	77.31 ± 1.1
MSCOCO-WikiArt Baseline	73.00	73.78	76.37	89.04	**78.05** ± 0.9

images, and a dataset of paintings mostly collected from WikiArt [119] as the source of style images. Our strategy, instead, avoids the use of extra datasets besides those directly involved in the domain generalization task as source domains. The reason is **twofold**: first, we want to keep the method as simple as possible, without the need to rely on external data; second, in order to perform a fair comparison with competing DG methods all of the algorithms should have access to the same source information.

Still, it may still be interesting to evaluate the difference between our approach and the use for style-transfer of the original AdaIN model trained on MSCOCO and WikiArt. We analyze this difference both at the qualitative and quantitative levels. Figure 4.4 shows an example of style-transfer obtained with the two strategies. Specifically we consider a dog image drawn from the PACS Photo domain and we analyze the images obtained by borrowing the style from the represented Art Painting guitar image. We compare the stylized sample produced with the MSCOCO-WikiArt AdaIN model against the outcomes of the four AdaIN variants obtained by training in turn on three

source domains and leaving the last as target.

As can be observed, the transformed images produced in output are not so different in terms of image quality. In Table 4.4 we report the results of our quantitative analysis, which compare the performance of our Stylized Baseline on PACS AlexNet with the analogous Baseline trained using the augmented data produced with the AdaIN MSCOCO-WikiArt pretrained model. The last one shows a slightly better accuracy which is though not significant if we consider the related standard deviation.

4.4 Conclusions

This chapter has focused on analyzing a disconnection between two branches of the research on DG. On one side there are solutions focusing on feature adaptation or proposing tailored training procedures that try to bridge the visual domain shift at the learning-level; on the other, data augmentation-based techniques that often adopt computationally expensive generative approaches to tackle the problem at the data-level. While the former group includes theoretically grounded approaches that are however often characterized by a complex design and that often guarantee *very slight* performance improvements, the latter is based on a much simpler idea which is often more effective, *i.e.* the fact that a larger and more varied training dataset leads to a more generalizable model. These two groups are often analyzed separately, as researchers proposing solutions belonging to the first group avoid comparing them with data-based approaches, deeming unfair a comparison with methods having access to an extended training dataset. The direct consequence is that it is often unclear which of the two approaches is better in general or in specific situations, and if their apparent orthogonality allows to combine them to further improve the generalization ability of trained models.

The purpose of our analysis was thus to recompose the literature, by proposing a very simple style-transfer-based data augmentation approach which we evaluated as an auxiliary augmentation strategy both on top of a naïf baseline and of a number of state-of-the-art methods for which we designed tailored integration strategies. The experimental analysis has pointed out that the performance of the considered methods improves over their original versions not including this augmentation, but surprisingly the same methods also lose their original effectiveness, not showing any improvement over the new data augmented baseline.

As other concurrent technical reports [47], our work suggests the need to shed new light on domain generalization, with a special focus on a call for novel strategies able to take advantage of the data variability introduced by cross-domain style-transfer. In recent years, this call has been at least partially answered through the presentation of many DG methods based on data augmentation strategies, as highlighted by an influential survey [190].

Chapter 5

Pushing the boundaries of distribution shift analysis

The focus of this chapter is on two research settings characterized by a multitude of challenges that have to be tackled all together in order to obtain dependable algorithms. In particular, among these challenges, there is both a visual distribution and a semantic shift, whose joint presence risks becoming an additional source of errors. The two settings studied are **multi-source open-set domain adaptation** and **cross-domain open-world recognition** and the two proposed solutions, called respectively **HyMOS** and **COW**, are connected not only because they are designed for problems characterized by multiple challenges, but also because they both tackle these multiple challenges with a **single contrastive-based learning objective**. This approach represents a paradigm shift with respect to the standard practice, consisting in the design of algorithms obtained as naive combinations of strategies designed to tackle individual challenges. We argue here that a much more robust approach can be obtained by designing a joint solution to all those individual challenges, and we propose to reach this result through contrastive learning thanks to the interesting structural properties of its learned feature space.

Part of the work described in this chapter has been previously published in two papers:

- [15] S. Bucci, F. Cappio Borlino, B. Caputo and T. Tommasi
  Distance-based Hyperspherical Classification for Multi-source Open-Set Domain Adaptation
  Winter Conference on Applications of Computer Vision, WACV 2022

- [17] F. Cappio Borlino, S. Bucci, and T. Tommasi
  Contrastive Learning for Cross-Domain Open World Recognition
  The 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2022)

5.1 The challenges of open-world learning

Vision systems are often trained in ***closed-world*** scenarios. This means that, in the learning phase, they experience a *narrow* portion of the world, which includes a limited set of semantics and a highly biased visual distribution. These models fail when presented with new environmental conditions, new data distributions, and novel classes at deployment time. This limited generalization ability can be at least partially explained considering those downsides of deep networks that we have described in the past, including the supervision collapse [36] and the tendency to provide overconfident predictions [118].

Moving to *open-world learning* is the only way to obtain robustness to these situations. However, how to do this is a long-standing research question. Indeed, besides the simple name, open-world learning is a complex problem which entails a number of challenges, among which a significant portion is caused by the fact that train and test data may be sampled from different distributions. For *vision* tasks, this means dealing with datasets showing significant differences in appearance and style among each other, but possibly also non completely overlapping sets of semantics, with novel classes appearing after deployment.

When a single problem involves multiple challenges, one of the most common strategy to deal with it consists in applying the *divide et impera* policy: **specific solutions are designed for specific aspects of the problem**, with an overall strategy to tackle all of them which can be obtained only by combining the aspect-specific solutions.
This approach has a number of **downsides**:

- when focusing on a specific challenge, it is necessary to make *strong assumptions* about the others. E.g.:
  - most cross-domain analysis scenarios are *closed-set*, which means that they assume train and test distributions to have matching label sets;
  - most open-set recognition studies are carried out in the *closed-domain* assumption, which means considering train and test distributions differing only in terms of semantics but not in terms of visual features;
- combining solutions defined for different aspect-specific challenges often leads to an overall complex approach, based on a multitude of learning objectives and a set of manually tuned hyperparameters which control their relationships.

We want here to propose a **different paradigm** to deal with some open-world research scenarios, consisting in adopting a **single learning objective** able to tackle multiple sub-problems at once. In order to reach this result we discard the standard supervised cross-entropy-based learning framework in favor of one based on **supervised contrastive-learning**. This learning objective, which provides performances similar to the cross-entropy when working in closed-world settings [66], **poses some**

constraints on the structure of the learned feature space, which helps in making models robust in open-world scenarios. In particular, the properties of its *hyperspherical* feature space encourage cross-domain alignment, while enabling rejection of unknown samples. We show how this result can be achieved in two open-world research settings, which, in spite of some differences in their formal formulation, are both characterized by the study of a joint occurrence of a visual domain and a semantic shift.

The contributions of this chapter are:

- the presentation and analysis of two open-world learning settings, namely **multi-source open-set domain adaptation** (MSOSDA) and **cross-domain open-world recognition** (CD-OWR). These two research problems are presented in detail, their challenges are highlighted, and the current state-of-the-art is presented;
- the proposal of **HyMOS** (Hyperspherical feature space for Multi-source Open-Set domain adaptation), an algorithm designed to tackle the MSOSDA problem by using a single contrastive learning objective, style-transfer for feature alignment, and statistics about the structure of the feature space to separate known and unknown samples;
- the proposal of **COW**, a strategy designed for CD-OWR, similar to HyMOS, but with the additional ability to support class incremental learning;
- an in detail quantitative analysis of these two methods, compared with previous state-of-the-art approaches, through large-scale comprehensive benchmarks specifically designed for the two research settings.

In the rest of this section we introduce the two research settings on which the chapter focuses and their related works.

5.1.1 Multi-source open-set domain adaptation

In many real-world learning scenarios dealing with *multiple sources* is more the rule than an exception, as the annotated data available to train a model may be the result of asynchronous multi-agent collection processes. This situation certainly represents a challenge, as it can result in a misalignment between the representations of different sources in a feature space which, as a consequence, will not support the definition of robust decision boundaries. At the same time, the availability of multiple sources can become a resource to support robust cross-domain performance if exploited in the correct way. Indeed, as we have seen in Chapter 4, the access to a set of source domains can be exploited to disentangle between domain-specific and domain-invariant features, leading to more robust models that focus on the latter to make decisions. This ability is fundamental when the visual domain met after deployment is different from

the training ones, even when it is known *a priori* thanks to the availability of an unlabeled set of samples collected from it. In this situation, feature alignment is one of the most common choices of strategy [115].

At the same time, a naïf feature alignment application may not always help, especially if the target set of available samples is really *unsupervised*, which means that it is an *uncurated* collection of samples coming from the target domain. Indeed, in this case, it may happen that the target set contains representatives of classes not present in the labeled source set, and, in the presence of a semantic shift of this kind, enforcing feature alignment may lead to *negative-transfer*, as alignment of domains does not necessarily involve alignment of classes [93].

This *multi-source open-set domain adaptation* scenario poses thus a multitude of challenges, as it is typical of many open-world learning settings: i) multiple labeled sources are available at training time, but they all come from different visual distributions which do not match the target one; ii) a set of unlabeled samples coming from the target distribution is also available to support adaptation, still, its truly unsupervised nature represents an additional challenge as this set could contain examples of unknown classes.

The difficulty of dealing with all these challenges at the same time pushes the researchers to focus only on specific aspects of the problem (*e.g.* multi-source closed-set domain adaptation, or single-source open-set one), and the availability of only one previous work focusing on this setting [130] highlights the need for the development of novel methods able to tackle all these challenges at once.

Our proposal exploits the power of contrastive learning and the properties of its hyperspherical feature space to correctly predict known labels on the target while rejecting samples belonging to any unknown class. ***HyMOS*** includes style-transfer among the instance transformations of contrastive learning to obtain domain invariance while avoiding the risk of negative-transfer. A self-paced threshold is defined on the basis of the observed data distribution and updates online during training, enabling known-unknown separation.

5.1.2 Cross-domain open-world recognition

Trustworthy autonomous agents should satisfy a number of requirements in order to be deployable in open-world contexts. Among these, there is the ability to recognize multiple objects in a variety of target conditions (*e.g.* change in viewpoint, camera equipment, illumination, weather, country), while correctly detecting samples belonging to novel categories. In many cases, however, this is not enough. For a practical example let's consider a home assistant robot: *i.e.* a robot designed to assist a human in carrying out a number of domestic tasks. In order to be really helpful, and adapt itself to the owner's needs, such a robot cannot be limited to recognizing only a pre-defined set of classes for which it was programmed: on the contrary, it should be able to incrementally learn to recognize new objects when its owner shows them to it, and then

to locate those objects when they are naturally arranged in different rooms, without getting confused by other objects for which it has not received instructions [69].

Indeed, the ability to evolve is fundamental for autonomous systems, whose knowledge cannot remain limited to the one injected by the manufacturer. However, this represents an additional challenge that needs to be tackled together with the others (*i.e.* domain and semantic shift), but that is usually studied as a problem in its own right. The setting that studies this aspect is called *Class Incremental Learning* (CIL): its focus is on extending an original model to accommodate novel classes in subsequent incremental tasks [132].

There have already been attempts of combining CIL with other settings: some works study *Open World Recognition* (OWR), which combines *open-set* learning, *i.e.* the ability to recognize a closed-set of categories while rejecting unknown samples, with CIL, but mainly disregard domain shift conditions [107, 105, 40]. However, a change in domain between training and test data can create confusion in the identification of the novel categories, and consequently, make their inclusion in the training process even more challenging.

The *Cross-Domain Open-World Recognition* setting (CD-OWR) has thus been proposed in order to provide a wider point of view on all the challenges that an autonomous agent has to face when deployed in an open-world setting [41]. This research problem, which can be seen as the union of OWR and Domain Generalization, still has to take hold, and no algorithm has been designed explicitly for it, with methods included in the first proposed benchmark [41] that are obtained as naïve combinations of approaches designed for sub-problems.

We thus propose to fill this gap by presenting **COW** (Contrastive Open-World), a variant of HyMOS, adapted to the CD-OWR setting. We show how a single supervised contrastive objective is suitable for open-world recognition while also promoting domain generalization. Specifically, in the hyperspherical feature space obtained via contrastive learning, samples of the same class tend to cluster together regardless of their domain, while novel categories appear in low-density regions. By considering the Nearest Class Mean [107] logic which is the basis of many OWR methods, it becomes clear that the described embedding is an ideal environment where a simple rejection rule can be applied on sample-to-prototype distances in order to identify novel categories. Moreover, our approach does not need class-specific rejection thresholds as the learned feature space pushes all clusters to have similar structures and distances, further simplifying the task. At the same time, the use of prototypes on an hyperspherical feature space enables decoupling the network output dimensionality from the number of classes that the network knows, and the chosen learning objectives naturally support the introduction of novel classes which are automatically accommodated by pushing existing class clusters apart.

5.1.3 Related works

Cross-domain learning and semantic shift

Most DA and DG works consider the exact **same class set shared by source and target**. This has started to change only recently with the emergence of the *open-set domain adaptatio*n setting [123, 93, 75, 14]. Much lower interest was shown in a similar *open-set domain generalization* setting, for which there is only one important work, which however focuses only on the multiple sources case [144].

Open-set domain adaptation Most of the studies focusing on domain adaptation consider the available target set to be *unlabeled*, but at the same time, they work under the *closed-set* assumption. This choice does not reflect a realistic data collection process. Indeed, with an uncurated target collection strategy, it's impossible to be sure that the target semantic content will perfectly match that of the source.

Open-Set domain adaptation tackles target domains which include new unknown classes with respect to the source. After the definition of the problem in [123], a first group of works focused on maximizing the separation between known and unknown target samples while exploiting adversarial-based methods to align the known classes [93]. More recently, different paradigms have started being proposed. In [75] a model is directly trained on the source with an extra set of negative samples produced via the suppression of class-specific feature activations. ROS [14] shows how to exploit the self-supervised rotation recognition task to deal with both feature alignment and known-unknown separation. PGL [102] exploits a graph neural network with episodic training to suppress the underlying conditional shift, while adversarial learning reduces the marginal shift between the source and target distributions. The only published method dealing with *multi-source open-set domain adaptation* is MOSDANET [130] which adds a clustering objective on top of a standard supervised classification model to maximize the similarity among samples of the same class but different domains. Moreover, it exploits adversarial learning for domain adaptation: a tailored margin loss penalizes cases with a small difference in known and unknown prediction output, while potential target samples are included in the training procedure via pseudo-labeling.

Universal Domain Adaptation The methods dealing with this setting cover a wide range of scenarios with private classes included in the source and/or the target set. In DANCE [139] a neighborhood clustering technique is integrated with the standard cross-entropy loss to learn the structure of the target, while an entropy-based score is used to align or reject the target samples. CMU [42] exploits a multi-classifier ensemble together with an unknown scoring function that combines entropy, confidence, and consistency measures.

Open-World Recognition

Hard-coded recognition skills are clearly not enough for autonomous robots that operate in unconstrained environments. *Class Incremental* (or *Continual*) *Learning* provides support with strategies that update pre-trained models by including new classes once they are observed. In order to be really useful, this should be done with as little access to the old data as possible (otherwise it would be more similar to a complete retraining), while also avoiding forgetting previous knowledge. One of the early incremental approaches was based on the Nearest Class Mean (NCM, [107]) classifier, which computes the per-class mean of the feature vectors of training samples, and each test sample is assigned to the nearest among these class prototypes. In the more recent literature, there are two main types of approaches. Some methods keep a small memory buffer of previous data in order to replay it while learning new classes [132]. Other more complex approaches do not require memory, but exploit extra objectives to avoid forgetting previous knowledge, one example being distillation [87]. The growing interest in this field is also testified by the ever-wider range of datasets designed for it [69, 98].

Dealing with unconstrained learning conditions means also not knowing a priori which classes will be encountered at test time. *Open Set Recognition* approaches are able to distinguish such unknown objects from the known ones while still classifying the known objects, and have been extensively studied by both the Computer Vision and robotics communities [180, 109]

Finally, *Open World Recognition* combines the two previous settings. It was first introduced in [6] which proposed NNO, a simple extension of the standard NCM strategy including an unknown rejection policy. More recent works are DeepNNO [105], the deep version of NNO, and B-DOC [40] that includes clustering objectives and class-specific rejection thresholds.

Contrastive Learning

Lately, self-supervised learning methods have shown that, by relying only on unlabeled data, it is still possible to get representation learning performance similar to those of the supervised approaches [49, 26, 24]. These models are all based on contrastive learning, a deep metric learning strategy that builds over instance discrimination techniques, treating every instance as a class of its own, and that aims at maximizing the agreement among multiple augmentations of the same sample while pushing different instances far apart. The effectiveness of the contrastive self-supervised learned embeddings is generally evaluated by using the trained feature extractor as the starting point for a downstream supervised task training. However, more direct ways to incorporate supervision have been attracting attention [66] lately, and show how view-invariance and semantic knowledge can be combined to tackle novelty detection [152], cross-domain generalization [186], and class incremental learning [104].

Learning on the Unit Hypersphere Fixed-norm representations have nice properties that support deep learning computational stability and their empirical success has been demonstrated over several tasks [163, 108]. In particular, [108] shows how setting class prototypes *a priori* on the unit hypersphere allows to free the output dimensionality from a constrained number of classes. The uniform distribution of the class centroids implies large margin separation among them and leaves space to include new categories while maintaining a highly discriminative embedding. A recent work has also highlighted how learning features uniformly distributed on the unit hypersphere with compact positive pairs is a crucial component of the success of contrastive learning [165].

5.2 Problem formalization

The two settings of *Multi-source open-set domain adaptation* and *Cross-domain open-world recognition* can be both described as designed to study open-world learning problems, *i.e.* research problems defined to study the challenges faced by CV systems deployed in the open-world.

Still, these two settings present some differences and is thus necessary to provide a formal definition for both.

5.2.1 Multi-source open-set domain adaptation

Formal problem definition:

- **data available at training time**:
  - n_S labeled source datasets $S = \{S_i\}_{i=1}^{n_S}$, each one consisting of a set of image-label pairs: $S_i = \left\{\left(\boldsymbol{x}_j^{(s)}, y_j^{(s)}\right)\right\}_{j=1}^{N_{S_i}} \sim p_i$. These sources share the same label set $y^{(s)} \in \mathcal{Y}_S$, but are sampled from different visual distributions $p_i \neq p_j,\ \forall i, j, i \neq j$;
  - an unlabeled target dataset $T = \{\boldsymbol{x}_i^{(t)}\}_{i=1}^{N_T} \sim q$, drawn from a different distribution: $q \neq p_i,\ \forall i = 1, \dots, n_S$. The target label set does not match the source one, $\mathcal{Y}_S \subset \mathcal{Y}_T$, but it contains additional classes $\mathcal{Y}_{T \setminus S}$ which are considered *unknown*
- **goal**: to correctly classify the samples in $T = \left\{\left(\boldsymbol{x}_i^{(t)}, y_j^{(t)}\right)\right\}_{i=1}^{N_T}$, by assigning them to the correct class if $y^{(t)} \in \mathcal{Y}_S$ or by marking them as *unknown*;
- **data available at inference time**: $\{\boldsymbol{x}_i^{(t)}\}_{i=1}^{N_T}$.

Starting from this setup it may be difficult to bridge the domain shift while avoiding the risk of *negative-transfer*, especially when the *openness* $\mathbb{O} = 1 - \frac{|\mathcal{Y}_S|}{|\mathcal{Y}_T|}$ increases.
Note: for simplicity, we have formalized the problem with a single target set, used at training time as an unsupervised source of information about the target domain and at inference time as test set. This *transductive* setting is not the only one possible, and a *non-transductive* one may equally occur, where the target data available at training time does not match with the test data, as described in the generic UDA case, see Sec. 2.2.2.

5.2.2 Cross-domain open-world recognition

The main differences with the previous setting are the fact that a single source dataset is available, that the target domain is not known at training time (*i.e.* this is a domain generalization setting, not a domain adaptation one), and most importantly that there are multiple training episodes:

- **data available at training time**:
  - **base episode**: $S_0 = \left\{ \left(\boldsymbol{x}_j^{(s)}, y_j^{(s)} \right) \right\}_{j=1}^{N_{S_0}} \sim p_0$, where $y^{(s)} \in \mathscr{Y}_{S_0}$;
  - **subsequent episodes** $S_i = \left\{ \left(\boldsymbol{x}_j^{(s)}, y_j^{(s)} \right) \right\}_{j=1}^{N_{S_i}} \sim p_i$ with $i = 1, \dots, K$ and $\mathscr{Y}_{S_i} \cap \mathscr{Y}_{S_j} = \varnothing,\ \forall i, j \in [0, \dots, K]$ and $i \neq j$. All source samples come from the same visual distribution: $p_i = p_j = p_S,\ \forall i, j$
- **goal**: after episode k, to correctly classify the samples in $T = \left\{ \left(\boldsymbol{x}_i^{(t)}, y_j^{(t)} \right) \right\}_{i=1}^{N_T} \sim p_T$, by assigning them to the correct class if $y^{(t)} \in \{\bigcup_{i=1}^{k} Y_{S_i}\}$ or by marking them as *unknown*. We have $p_T \neq p_S$, and $\{\bigcup_{i=0}^{K} \mathscr{Y}_{S_i}\} \subset \mathscr{Y}_T$
- **data available at inference time**: $\{\boldsymbol{x}_i^{(t)}\}_{i=1}^{N_T}$

The classes included in S_0 are called *base classes*, as they are used to build an initial knowledge base. Each incremental task (episode) has its own class set which does not overlap with the previous ones. We call *source domain* the entire labeled training set $S = \{\bigcup_{k=0}^{K} S_k\}$ that is drawn from data distribution p_S. The unlabeled test set T is not seen during training and is drawn from the target distribution p_T, with $p_S \neq p_T$.

5.2.3 Contrastive Learning formulation

We adopt a contrastive learning formulation that takes inspiration from the self-supervised literature but includes supervision [66] in the learning objective. The rationale behind this choice is that supervised contrastive learning imposes some constraints on the structure of the learned feature space, constraints that are not shared with the standard supervised learning paradigm, *i.e.* cross-entropy loss.

In its training procedure, **self-supervised** contrastive learning [26, 49] takes two augmented views of each input sample and propagates them through a feature encoding network. The views are obtained via standard augmentation strategies such as *grayscale*, *random crop*, and *color jittering*. All the encoded features $Enc(\boldsymbol{x}_k^{(s)})$ in the double batch $B = \{k = 1, \dots, 2K\}$ enter then the contrastive head that further projects them to a normalized embedding, producing $\boldsymbol{z}_k^{(s)} = Proj(Enc(\boldsymbol{x}_k^{(s)}))$. On the obtained

hyperspherical space, the samples are compared among each other: the training objective consists in maximizing the similarity between the representations of the augmented views of the same instance (positive pairs), while minimizing that among the representations of different instances (negative pairs).

In the **supervised scenario**, when semantic labels are available, the positive pairs are identified considering the class label [66]. This means that given an *anchor* representation with index k, all the representations of samples belonging to the anchor's category can be used to build positive pairs, and all the others to build negative pairs. We indicate with $\nu(k) = B \setminus \{k\}$ the double batch without the *anchor* of index k: the positive indices are $\pi(k) = \{k' \in \nu(k) : y_{k'}^{(s)} = y_k^{(s)}\}$. The supervised contrastive loss function [66] is:

$$\mathscr{L}_{supcon} = \sum_{k=1}^{2K} \frac{-1}{|\pi(k)|} \sum_{k' \in \pi(k)} \log \frac{\exp(\sigma(\boldsymbol{z}_k^{(s)}, \boldsymbol{z}_{k'}^{(s)})/\tau)}{\sum_{n \in \nu(k)} \exp(\sigma(\boldsymbol{z}_k^{(s)}, \boldsymbol{z}_n^{(s)})/\tau)}, \tag{5.1}$$

where $\tau \in \mathbb{R}^+$ is a parameter called temperature, and $\sigma(\cdot, \cdot)$ is the cosine similarity.

Adopting this learning objective ensures that, in the final feature space, the representations of samples of the same class will be well clustered and far from representations belonging to different categories. The overall structure is thus a hypersphere with compact and well-separated class clusters placed on its surface.

Table 5.1: Comparison with existing open-set and universal domain adaptation approaches. HPs indicate the hyperparameters, $|\mathcal{Y}_s|$ the number of source categories, $|S|$ is the number of source domains. Note that synthesizing new samples is a time-consuming operation and any validation procedure requires at least a dedicated per-dataset tuning.

Method	No. of Losses	No. of HPs	Threshold
Inheritable [75]	4	2	not used - synthesize *unknown* target
ROS [14]	6	4	reject a fixed portion of Target
CMU [42]	$2 + \|\mathcal{Y}_s\|$	3	validated
DANCE [139]	3	3	fixed value depending on $\|\mathcal{Y}_S\|$
PGL [102]	3	4	reject a fixed portion of Target
MOSDANET [130]	$4 + \|S\|$	2	validated
HyMOS	1	1	self-paced, updates online while training

5.3 Contrastive learning for multi-source open-set domain adaptation

5.3.1 Preliminaries

In order to tackle *multi-source open-set domain adaptation* we aim at building a robust, highly structured feature space with domain-aligned, compact, and well-separated class clusters, where *unknown* target samples lay away from clusters' centroids. We reach this result by minimizing the supervised contrastive loss [66] and by paying particular attention to how data is fed to the model.

Our strategy is based on:

- a domain- and class-balanced **sampling approach for mini-batch definition**. Its purpose is to exploit the peculiarities of our learning objective in order to obtain class-wise alignment among the different source domains;
- the inclusion of **style-transfer** among the standard **semantic-preserving transformations** used to build sample pairs in contrastive learning. Taking this measure explicitly forces the model to ignore style-specific features in the learning phase, effectively leading to learning domain-invariant representations;
- a **refinement of source-target alignment** obtained by progressively including the target domain in the learning objective through **self-training;**
- the adoption of a **self-paced threshold for *known-unknown* separation** which depends directly on the learned data distribution and thus adapts automatically to

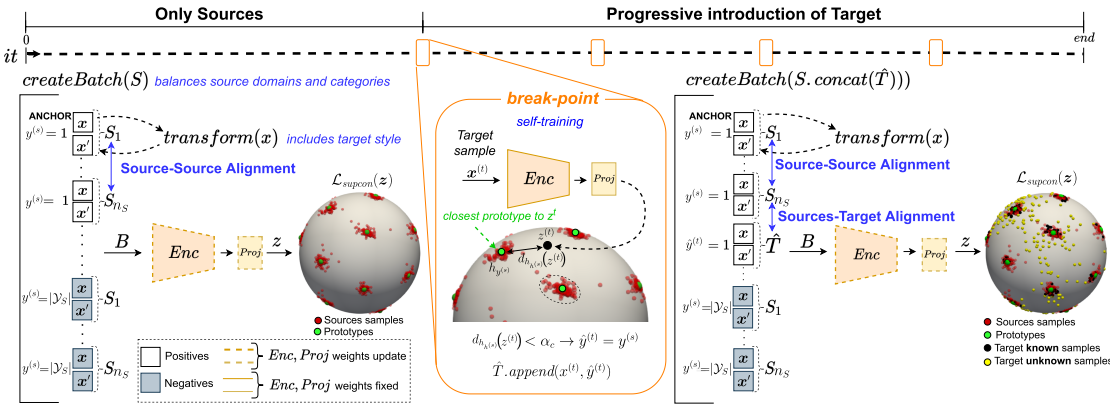


Figure 5.1: Schematic illustration of HyMOS (best viewed in color). We use the same notation adopted in Algorithm 3, please refer to it to follow the flow of the method.

different tasks. This threshold is used both at inference time and during training, to select *known* target samples for self-training.

The overall algorithm is called ***HyMOS*** which stands for *Hyperspherical feature space for Multi-source OpenSet domain adaptation.* All its components take part and contribute to the definition of a method that tackles all the challenges of the studied problem by exploiting a single learning objective. This *keep-it-simple* design approach to address such a complex problem deeply contrasts with the paradigm followed by most of the alternative methods, which relies on decomposing the problem, designing different modules and loss functions to address different challenges, and obtaining a complete solution only by combining these modules through a multitude of hyperparameters and making choices based on hard-to-validate heuristics. We summarize this situation in Tab. 5.1.

We focus now on presenting the components of HyMOS in detail. An overview is also illustrated in Figure 5.1 and summarized in Algorithm 3, whereas Algorithm 4 focuses on the test-time procedure.

5.3.2 Sampling approach for mini-batch definition

The supervised contrastive loss aims at learning compact class clusters with large margins. We exploit this ability to obtain source-source class-wise domain alignment by designing a specific procedure to build training mini-batches with samples coming from different visual domains. In particular, we evenly divide each batch to cover all source classes $\mathcal{Y}_S$, and, for each of them, we select an equal number of samples from all the n_S source domains. The loss function of Eq. 5.1 takes care of the rest, providing an embedding space where representations of samples of the same class are grouped in the same region regardless of the domain, while different classes are kept far apart from each other.

5.3.3 Style-transfer as part of contrastive learning

In the self-supervised contrastive learning pipeline a fundamental role is played by the semantic-preserving transformations used to obtain augmented views of input: they are meant to force the model to focus on core semantic information while becoming invariant to the irrelevant cues that these transformations introduce. When dealing with data from different domains we desire a representation that is able to neglect *major* differences in visual appearance that go beyond *grayscale* or *color jittering*. This calls for dedicated semantic-preserving image transformations. We thus propose to extend the original augmentation pipeline by the introduction of an additional transformation based on style-transfer: this is perfectly suitable for our goal as it does not affect the image content while modifying significantly the global image texture. Similarly to what we proposed in Chapter 4, we adopt for this process an AdaIN [55] model, that we trained jointly on source and target data. In this case we want to learn how to transfer the style from target images into source ones. As this augmentation is applied randomly, during learning the supervised loss function will explicitly compare original source images with target-like ones and learn to ignore the style difference.

One of the main advantage of this strategy to obtain style invariance is that it is safe from *negative-transfer*. This is one of the main issues usually faced in open-set domain adaptation and sparks from the risk of aligning *unknown* target categories with *known* source ones. As a result, the existing methods [14, 42, 139, 93] usually try to mitigate the problem by directly avoiding the inclusion of unknown samples in the adaptation process. However, this strategy depends on a correct identification of the unknown samples even before the learning of a domain invariant model, which is clearly a complex task. With style-transfer, instead, we learn a domain-agnostic representation since the beginning of the training process, and given that this transformation disregards the semantic content of the style image we can safely draw the style also from samples belonging to *unknown* categories.

5.3.4 Domain alignment refinement via self-training

According to what we have seen so far, if target labels were available during training, it would be possible to obtain a *perfect* source-target alignment by simply including the target set as an additional source in the described learning pipeline. Of course, this is not the case. Still, once the model trained on style-transfer-enriched source data is robust enough, one could use it to produce pseudo-labels for target data by simply exploiting its predictions. We follow this approach in an episodic fashion: the first step is a source-only learning episode, after which we start progressively including the target samples in our learning objective. In particular, we perform periodic evaluation steps that we call *self-training break-points*. These allows us to select target samples which are confidently recognized as *known*. Through this iterative technique, we propagate label knowledge from source to target data, improving the compactness of our class

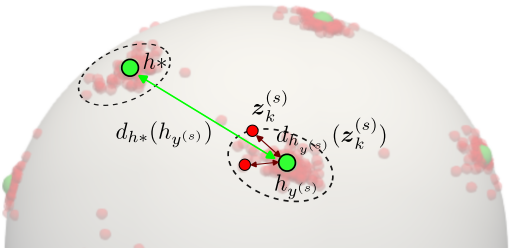


Figure 5.2: Illustration of the distances used for the class prediction and the self-training procedure.

clusters while progressively leaving *unknown* target data in low-density regions of our hyperspherical feature space.

5.3.5 Semantic discrimination on the Hypersphere

The described learning procedure enables obtaining an embedding in which *known* class clusters are compact and separated by large margins, while *unknown* samples are left isolated. This embedding provides the ideal condition to perform distance-based classification, in a NCM fashion [107]. It should be noticed that this strategy to use the feature space built via contrastive learning is fundamentally different from the one adopted in previous literature [66, 152], where the contrastive objectives were used only as pretext learning tasks and the projection head was later dropped before fine-tuning the model with a standard cross-entropy loss (a procedure sometimes called Guillotine Regularization [10]). In our case, we keep the projection head and exploit its hyperspherical output for delivering the final predictions. In particular, for each source class $y^{(s)}$, we compute a prototype by re-projecting on the unit hypersphere the feature average of the training samples belonging to it:

$$h_{y^{(s)}} = \frac{\frac{1}{N_{y^{(s)}}} \sum_{k \in y^{(s)}} \boldsymbol{z}_k^{(s)}}{\left\| \frac{1}{N_{y^{(s)}}} \sum_{k \in y^{(s)}} \boldsymbol{z}_k^{(s)} \right\|_2} \tag{5.2}$$

In the evaluation phase, for any target sample $\boldsymbol{z}^{(t)}$ we measure its cosine similarity σ with each source class prototype and we rescale it in $[0,1]$ in order to define the distance metric that we use as a confidence measure for label assignment:

$$d_{h_{y^{(s)}}}(\boldsymbol{z}^{(t)}) = \{1 - \sigma_{[0,1]}(\boldsymbol{z}^{(t)}, h_{y^{(s)}})\} \quad \text{for } y^{(s)} \in \mathcal{Y}_S$$

In this context of distance-based classification, in order to decide whether a sample belongs to a *known* category or to an *unknown* one we need to define a threshold on

the distance between this sample and the nearest *known* class prototype. The definition of this threshold is not easy and it's therefore a widely discussed problem in the open-set DA literature, with many methods choosing values a priori and keeping them fixed while training [42, 139]. In contrast, our proposal is to define it as a **function of the observed data distribution**, so that its value can change online during training and adapt to different tasks. In order to obtain this result we define two metrics that summarize a description of the data distribution. We start from the ***class sparsity***:

$$\theta = \frac{1}{|\mathscr{Y}_S|} \sum_{y^{(s)} \in \mathscr{Y}_S} d_{h_*}(h_{y^{(s)}}) \tag{5.3}$$

where h_* is the closest prototype to each $h_{y^{(s)}}$. This metric is the average of the prototype-to-prototype minimal distances and provides a measure of inter-class separation. The second metric is the ***class compactness***:

$$\phi = \frac{1}{|\mathscr{Y}_S|} \sum_{y^{(s)} \in \mathscr{Y}_S} \left\{ \frac{1}{N_{y^{(s)}}} \sum_{k \in y^{(s)}} d_{h_{y^{(s)}}}(z_k^{(s)}) \right\} \tag{5.4}$$

which evaluates whether the samples of each class are grouped tight around the corresponding prototype (see Figure 5.2).

Given these metrics, we can easily notice that a training performed on a dataset with a large number of categories, each with small intra-class variability, results in a feature scenario with high compactness but low sparsity, for which a low threshold is needed. On the other extreme, a training performed on a dataset with a limited number of categories showing large intra-class variability corresponds to a low compactness and high sparsity condition, for which a higher threshold may be preferred.

We thus compute our threshold by:

$$\alpha = \phi \cdot \left[log\left(\frac{\theta}{2\phi}\right) + 1 \right] , \tag{5.5}$$

where $\frac{\theta}{2\phi}$ estimates the average ratio between the distance of two adjacent prototypes and the radii of the respective clusters. The application of this kind of threshold at inference time is straightforward:

$$\hat{y}^{(t)} = \begin{cases} \underset{y^{(s)}}{\text{argmin}}\ (d_{h_{y^{(s)}}}(\boldsymbol{z}^{(t)})) & \text{if } \min_{y^{(s)}}(d_{h_{y^{(s)}}}(\boldsymbol{z}^{(t)})) < \alpha \\ \text{unknown} & \text{if } \min_{y^{(s)}}(d_{h_{y^{(s)}}}(\boldsymbol{z}^{(t)})) \geq \alpha \end{cases} \tag{5.6}$$

The same threshold is applied also for the self-training *break-points* phases described before. In this case, however, it is important to be particularly cautious, as a wrong classification of unknown samples as known may lead to error propagation through the self-training step. We thus include a multiplier α_m that allows us to keep a more conservative threshold: $\alpha_c = \alpha_m \cdot \alpha$. This multiplier is fixed to 0.5 and it is the only hyperparameter of HyMOS.

5.3.6 Implementation details

We implement HyMOS with an architecture composed of an *encoder* and a *contrastive head*, where the former is a standard ResNet-50 backbone, while the latter is a simple MLP composed of two fully connected layers (of size 2048 and 128) The network is trained end-to-end by minimizing the supervised contrastive loss of Equation 5.1, where we set $\tau = 0.07$ as in [152]. Our final distance-based classification strategy is applied in the hyperspherical space produced by the model. Given that the output dimension is not constrained by the number of classes the architecture remains exactly the same for all our experiments.

We initialize the backbone network with an ImageNet1k pre-trained SupClr model [66] and train HyMOS for 40k iterations with the balanced data mini-batch definition strategy described before. We use a linear warm-up schedule for the learning rate, starting from 0 and going up to 0.05 at iteration 2500, after which we transition to cosine annealing, decreasing the learning rate back to zero through the rest of the training. We use LARS as optimizer [178], with momentum 0.9 and weight decay 10^{-6}.

As mentioned in Sec. 5.3.4, the training is performed in an episodic fashion. The first 20k training iterations are necessary to build a dependable model that can be later used for self-training. As a result, in this first training stage, the target data is used exclusively for the style-transfer-based data augmentation. After 20k iterations we start performing evaluation steps that we call self-training *break-points*, one every 5k iterations. After each *break-point* we include in the training dataset the target samples classified as known with a confidence higher than α_c. This procedure allows us to progressively include more and more target data in the training dataset.

5.3.7 Experimental protocol

Datasets In order to evaluate our approach we adopt the benchmark proposed in [130], that is built on top of three image classification datasets in which the set of available domains is split into sources and target, with a single domain considered in turn as target.

Office31 [136] comprises three domains: Webcam (W), Dslr (D), and Amazon (A) each containing 31 object categories. We set as known the first 20 classes in alphabetic order, while the remaining 11 are considered unknown.

Office-Home [159], already mentioned in Chapter 4, is composed of four domains: Art (Ar), Clipart (Cl), Product (Pr), and RealWorld (Rw) with 65 classes. The first 45 categories in alphabetic order are considered known, and the remaining 20 unknown.

DomainNet [127] is a significantly more challenging testbed than the previous ones: it contains six domains and 345 classes. We consider only the domains Infograph (I), Painting (P), Sketch (S), and Clipart (C), selecting randomly 50 samples per class or using all the images in case of lower cardinality. The first 100 classes in alphabetic order are known, while the remaining 245 are unknown.

Table 5.2: Results averaged over three runs for each method on the DomainNet, Office31, and Office-Home datasets.

	Method		DomainNet			Office31				Office-Home				
			→ S	→ C	**Avg.**	→ W	→ D	→ A	**Avg.**	→ Rw	→ Cl	→ Ar	→ Pr	**Avg.**
HOS	Source Combine	Inheritable [75]	34.8	44.0	39.4	76.6	79.5	70.0	75.4	63.2	52.6	48.7	60.7	56.3
		ROS [14]	44.5	52.4	48.5	81.8	80.1	64.7	75.5	**73.0**	57.3	61.6	69.1	65.3
		CMU [42]	38.1	35.5	36.8	61.4	64.0	56.4	60.6	70.8	50.0	58.1	69.3	62.1
		DANCE [139]	30.0	37.6	33.8	38.5	59.7	58.0	52.0	12.4	16.1	18.6	22.9	17.5
		PGL [102]	18.5	19.4	19.0	43.3	37.7	35.6	38.9	40.0	31.5	31.8	42.2	36.4
	Multi-Source	MOSDANET [130]	40.0	39.3	39.6	60.5	71.5	**73.9**	68.6	65.0	51.1	54.3	65.9	59.1
		HyMOS	**57.5**	**61.0**	**59.3**	**90.2**	**89.9**	60.8	**80.3**	71.0	**64.6**	**62.2**	**71.1**	**67.2**
AUROC	Source Combine	ROS [14]	63.9	68.0	66.0	93.9	95.2	**73.5**	87.5	80.8	69.6	73.7	79.4	75.9
	Multi-Source	**HyMOS**	**71.9**	**75.8**	**73.9**	**96.9**	**96.1**	71.0	**88.0**	**81.1**	**76.4**	**75.3**	**79.6**	**78.1**

Competitors We compare HyMOS with several state-of-the-art baselines proposed for single-source open-set (Inheritable [75], ROS [14], PGL [102]), multi-source open-set (MOSDANET [130]), and universal domain adaptation (CMU [42], DANCE [139]).

For all these baseline methods the original implementations are publicly available, with the only exception of MOSDANET [130] for which we obtained the code via private communications with the authors. As a consequence, for all the experiments we use the codebases provided by the original authors. For the methods that do not specify how to manage multiple sources, we simply combine all of them building a single source dataset (*Source Combine*).

Performance metrics For a fair comparison, we adopt the **HOS** evaluation metric as described in Sec. A.4.2 and defined in Equation A.4.

5.3.8 Experimental results

Main benchmark

In Table 5.2 we collect our evaluation results, which show that HyMOS outperforms all the baselines. Its gain with respect to the best competitor ROS, varies from 1.9% points on OfficeHome, up to 10.8% on DomainNet. Besides being simpler than the reference approaches, HyMOS proves to be robust to the significantly different scenarios covered by the three datasets in terms of the number of shared and private classes, as well the as nature and extent of the domain gap. These peculiarities make HyMOS the most suitable approach for a variety of real-world applications.

We also benchmark HyMOS, against the best competitor ROS, in terms of the **AUROC** out-of-distribution detection metric (see Sec. A.3.2). This metric is relevant as, thanks to its threshold-independent nature, it us allows to disentangle the quality of the normality evaluation function from the one of the thresholding strategy.

In our case, the *normality score* used to evaluate whether a sample is *known* or *unknown* is its distance from the nearest source class prototype, while ROS exploits a combination of entropy and probability output of an auxiliary rotation recognition

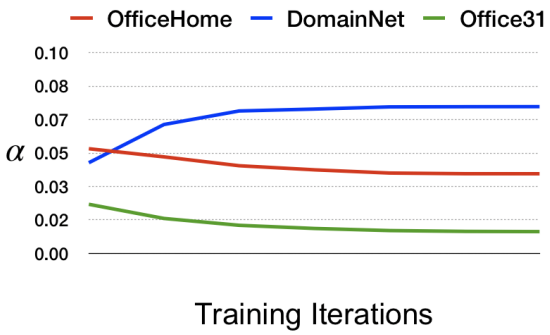


Figure 5.3: Analysis on the dynamic threshold α at different training iterations.

classifier. Even in this case HyMOS significantly outperforms ROS, proving that its better HOS results do not spark only from a sane thresholding approach, but are also a direct consequence of the high quality of the learned hyperspherical embedding space.

Analysis on the threshold

As described in Sec. 5.3.5 the dynamic threshold α is computed as a function of the data distribution. In Figure 5.3 we provide an overview of α value at different training iterations: for Office31 and Office-Home the value decreases over time while for DomainNet it increases. These different trends evidence how the data clusters move: as the training proceeds they become more compact and their reciprocal distance increases towards a more uniform class distribution on the hypersphere. For DomainNet the second event occurs faster than the first: this trend is correlated with the number of classes which is higher with respect to that of other datasets. In all cases the threshold converges to a stable value towards the end of the training.

The α_m multiplier used as part of the *break-point* evaluation strategy to compute a *conservative* threshold is the only hyperparameter of HyMOS: its goal is to have a high *precision* in the recognition of *known* classes, even if the *recall* may be low. Table 5.3 shows that $\alpha_m = 0.5$ is a safe choice regardless of the dataset. Moreover, by tuning this multiplier, the HOS performance of HyMOS remains always competitive with ROS, and can even increase as in the case of DomainNet for $\alpha_m = 1$.

Increasing the Openness Level

In many real-world scenarios, it is not possible to have direct control over the number of *unknown* classes in the unlabeled target, and it is natural to expect more *unknown* categories than *known* ones. In order to study how HyMOS reacts to different openness levels, we consider the DomainNet dataset and exploit its large class cardinality. The plot in Figure 5.4 shows the HOS accuracy of HyMOS and how it outperforms its best competitor ROS at different openness values $\mathcal{O} \in \{0.5, 1\}$.

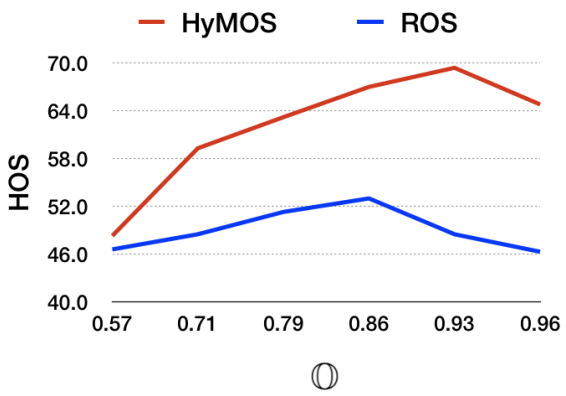


Figure 5.4: Performance of HyMOS and ROS [14] at different openness ($\mathcal{O}$) levels.

Table 5.3: Average performance (HOS) when changing the train-time multiplier α_m for the self-paced threshold α.

Method		**DomainNet**	**Office31**	**Office-Home**
HyMOS	$\alpha_m = 0.3$	55.1	79.2	65.8
	$\alpha_m = 0.5$	59.3	**80.3**	**67.2**
	$\alpha_m = 0.7$	60.8	78.2	66.8
	$\alpha_m = 1.0$	**61.4**	74.1	65.8
ROS [14]		48.5	75.7	65.3

5.3.9 Ablation Analysis

The main idea behind the development of HyMOS is that a robust method designed for the multi-source open-set domain adaptation task should be able to tackle all of this problem's challenges at once.

In the following, we focus on each of these challenges, providing a detailed ablation analysis that sheds light on the inner functioning of our method. The results of this analysis are collected in Table 5.4.

Source-Source Alignment

As widely proved in the domain generalization vast literature (see Sec. 2.2.1), reducing the domain shift among the available sources improves model generalization. For this reason, a dedicated source alignment component is included in the only existing multi-source Open-Set method MOSDANET [130].

In the HyMOS training phase, cross-source adaptation is obtained by combining the supervised contrastive learning loss with an accurately designed balanced batch sampling strategy, thanks to which the learning objective provides a strong class-wise alignment by, regardless of the domain, pulling together samples of the same class and pushing away samples of different classes. HyMOS shows a gain in performance of

Table 5.4: Ablation Study, HOS results.

Method	Office-Home				
	→ Rw	→ Cl	→ Ar	→ Pr	**Avg.**
HyMOS	71.0	64.6	62.2	71.1	**67.2**
w/o Source Balance	69.2	58.4	60.6	70.2	64.6
Style Tr. Target Known (Oracle)	70.7	63.7	62.5	71.2	67.0
w/o Style Transfer	69.5	56.4	60.0	68.3	63.6
w/o Self-Training	72.2	55.0	58.6	71.5	64.3
Improved Cross-Entropy	61.5	61.2	58.1	57.1	59.5
ROS [14]	73.0	57.3	61.6	69.1	65.3
+ Source Balance	75.2	55.5	62.6	66.9	65.0
+ Style Transfer	62.6	46.3	52.0	60.1	55.2
+ Self-Training	69.6	59.1	61.5	60.5	62.7
+ S. Balance, Style Tr., Self-Train.	62.0	40.4	52.2	62.4	54.3

2.6% over its version without this balancing strategy (see row *w/o Source Balance*).

Source-Target Adaptation

HyMOS obtains source-target alignment through the joint work of two of its components: the style-transfer-based augmentation strategy and the auto-regulated progressive self-training procedure.

We highlight the importance of both components:

- the style-transfer-based data augmentation approach allows to bootstrap the model's style-invariance without incurring in *negative-transfer*. This is proved by the negligible difference between HyMOS's results and those in the table row *Style Tr. Target Known (Oracle)*, for which an Oracle version of our approach is used to extract target style only from *known* target categories. Moreover, the effectiveness of this augmentation approach is proved by the performance drop that is obtained if it gets disabled (row *w/o Style Transfer*);

- the self-training procedure further improves source-target alignment by integrating *known* target samples in the learning procedure effectively obtaining an increase in compactness of the known class clusters. Also in this case, disabling this strategy involves a significant performance drop (row *w/o Self-Training*).

Comparison with improved baselines

Source balance, style-transfer, and self-training appear as simple strategies that can be combined with any supervised learning model to improve its effectiveness in the multi-source open-set domain adaptation scenario. Still, we state that leveraging supervised

contrastive learning and its related hyperspherical embedding is crucial for the effectiveness of these strategies in the task at hand. To support our claim we substitute the contrastive loss of HyMOS with the standard cross-entropy loss. The row *Improved cross-entropy* reports the obtained results, showing that this baseline approach is significantly worse than HyMOS.

We perform a similar exercise by enriching our best competitor ROS [14] with source balancing, style transfer, and self-training:

- the source-balancing (row *+ Source Balance*) does not provide any improvement for ROS, mainly because this method is based on the cross-entropy loss which does not have the same clustering effect as the supervised contrastive one;
- the style-transfer produces a drop in performance (row *+ Source Balance*): by checking the non-aggregated predictions it is possible to observe a slight advantage in the recognition accuracy of the *known* classes, but a significant drop in the *unknown* accuracy which causes a decrease in the overall result. Once again this is probably due to the different behaviors between the two learning objectives;
- the self-training, implemented following [130], produces again a performance drop, which in this case is probably caused by an error propagation phenomenon induced by the cross-entropy well-known overconfidence issue [118].

Finally, when applying all the strategies at once, the results are similar to those obtained with style transfer alone. This last technique clearly steers the whole method towards a low performance.

Algorithm 3: HyMOS training procedure

Input: α_m, AdaIN model
Data: $\{\boldsymbol{x}^{(s)}, y^{(s)}\} \in S$, $\boldsymbol{x}^{(t)} \in T$
Output: Enc, $Proj$

Function `transform(`$\boldsymbol{x}$`):`
  $styleAugment = random(True, False)$
  $\boldsymbol{x}' = randomCrop(\boldsymbol{x})$
  if $styleAugment$ **then**
    return $styleTransf(\boldsymbol{x}')$ ▷ target style
  else
    return $grayScale(jitter((\boldsymbol{x}')))$
  end
Function `createBatch(`S`):`
  $batch = [\,]$ ▷ balance domains and categories
  for ***each*** $y^{(s)}$ ***in*** $\mathcal{Y}_S$ **do**
    for ***each*** S_i ***in*** S **do**
      $\boldsymbol{x}'_{(y^{(s)},S_i)} = transform(\boldsymbol{x}_{(y^{(s)},S_i)})$
      $batch.append(\boldsymbol{x}_{(y^{(s)},S_i)}, \boldsymbol{x}'_{(y^{(s)},S_i)})$
    end
  end
  return $batch$ ▷ $len(batch) = |\mathcal{Y}_S| \times |S| \times 2$
Function `main()`:
  $\hat{T} = [\,]$
  for it ***in*** $range(0, end)$ **do**
    if it ***in*** $break\text{-}points$ **then**
      $\hat{T} = [\,]$
      $\alpha \leftarrow$ (Eq. 5.5) ; $\alpha_c = \alpha_m \cdot \alpha$
      for $\boldsymbol{x}^{(t)}$ ***in*** T **do**
        $\boldsymbol{z}^t = Proj(Enc(\boldsymbol{x}^t))$
        $h_{y^{(s)}} \leftarrow$ `closest prototype to` $\boldsymbol{z}^{(t)}$
        if $d_{h_{y^{(s)}}}(\boldsymbol{z}^{(t)}) < \alpha_c$ **then**
          $\hat{y}^{(t)} = y^{(s)}$; $\hat{T}$.append$((\boldsymbol{x}^{(t)}, \hat{y}^{(t)}))$ ▷ self-training
        end
      end
    end
    $B = createBatch(S.concat(\hat{T}))$
    $\boldsymbol{z} = Proj(Enc(B))$
    $loss = SupClr(\boldsymbol{z})$ (Eq. 5.1)
    Update $Enc, Proj \leftarrow \nabla loss$
  end

Algorithm 4: HyMOS evaluation procedure

Input: Enc, $Proj$
Data: T
Output: Predictions on T

$\alpha \leftarrow$ Eq. 5.5
for ***each*** $\boldsymbol{x}_t$ ***in*** T **do**
    $\boldsymbol{z}^{(t)} = Proj(Enc(\boldsymbol{x}^{(t)}))$
    $h_{y^{(s)}} \leftarrow$ `nearest prototype to` $\boldsymbol{z}^{(t)}$
    if $d_{h_{y^{(s)}}}(z^{(t)}) < \alpha$ **then**
        $\hat{y}^{(t)} = y^{(s)}$
    else
        $\hat{y}^{(t)} =$ unknown

Table 5.5: Comparison with existing OWR (OS+CIL), DG and CIL approaches. HPs indicate the hyperparameters.

Method	No. of Losses	No. of HPs	Open-Set Recognition	Domain Generalization	Class Incremental Learning
NNO [6]	1	1	✓		✓
DeepNNO [105]	2	1	✓		✓
B-DOC [40]	3	2	✓		✓
SS-IL [2]	2	0			✓
RR [97]	2	1		✓	
SC [56]	1	1		✓	
RSDA [161]	2	1		✓	
SagNet [117]	3	2		✓	
COW	1	2	✓	✓	✓

5.4 Contrastive learning for cross-domain open-world recognition

5.4.1 Preliminaries

As was the case of MSOSDA, the cross-domain open-world recognition (CD-OWR) task involves a multitude of challenges: cross-domain learning (*i.e.* domain generalization), open-set learning, and incremental learning. These challenges are usually tackled individually, with complete strategies that are built by combining challenge-specific solutions. Indeed, this is the case for all the algorithms included in the first benchmark designed to study this problem [41]. As we believe that this approach is not robust, we propose once again a paradigm shift, in particular by designing an algorithm that, by exploiting a single learning objective, tackles all of the problem's challenges at once. Similarly to what we have done before, we reach this result by relying on the supervised contrastive loss function, and we take inspiration from this choice to choose a name for our method, which we call **COW**, standing for *Contrastive Open-World*. We summarize a comparison between the structures of COW and the other methods used for CD-OWR in Table 5.5.

COW takes vast inspiration from HyMOS, but includes some additions specifically designed for the incremental nature of the CD-OWR task, in particular a **tailored stopping criterion** for the **incremental learning protocol**, and a modified **thresholding strategy**. For both these components we rely on the stats about the structure of the feature space introduced for HyMOS: the *class sparsity* θ (Eq. 5.3) and the *class compactness* ϕ (Eq. 5.4).

5.4.2 Incremental learning protocol

When performing incremental learning, the greatest risk is to run into the phenomenon called *catastrophic forgetting*, *i.e.* the *forgetting* of the knowledge learned up until task $t-1$, when performing the training on task t. Complex approaches have been proposed to prevent this from happening, in our case, however, we want to keep the focus on the effectiveness of the chosen learning objective, thus we adopt a very simple incremental protocol based on two principles:

- as done by other incremental learning algorithms we keep a (limited- and fixed-size) **replay buffer** containing a subselection of samples from previous tasks;
- we perform **class-balancing** at the batch level, by putting in each training mini-batch at least two samples of each class. In this case, class-balancing is particularly important as it enables balancing novel and old classes, pushing for learning without forgetting.

We expect our learning objective to manage the available data and progressively make room on the hyperspherical feature space to accommodate new classes while exploiting replay samples to maintain the space reserved for the old ones.

Stop-training criterion

Intuitively, class clusters in our feature space cannot be well separated if $\theta < 2\phi$. Indeed, ϕ can be considered as a measure of the radius of clusters, thus if the distance between two class centroids is lower than the sum of their radii the two clusters will inevitably overlap. This situation should be avoided as, with overlapping regions, the samples of the two classes cannot be distinguished, but avoiding an overlap may not be enough: we want some *empty space* between class clusters to accommodate unknown data.

In order to reach this result, we impose a constraint on the quality of the structure of the feature space for our output model. We obtain this result by exploiting the fact that the compactness and separation of known class clusters increase during training, which allows us to enforce the described relation between θ and ϕ by using a specific stopping criterion in the learning procedure. In particular, we consider each learning task as converged (and thus finished) only when

$$\lambda > 1 + \varepsilon, \quad \text{with} \quad \lambda = \frac{\theta}{2\phi} \quad \text{and} \quad \varepsilon \geq 0 \tag{5.7}$$

Here ε can be seen as a *minimum desired margin* between two clusters, and is one of the two hyperparameters of our method.

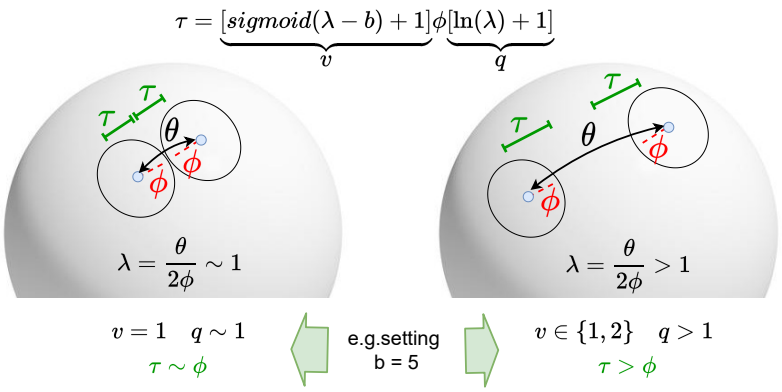


Figure 5.5: Visualization of two data clusters on the learned hyperspherical embedding and of the corresponding threshold value. Tuning b means tuning v in $\{1,2\}$

5.4.3 Threshold definition

By adopting the NCM [107] logic, each test sample is processed by computing its distance from the known class prototypes. This approach is particularly suitable in the OWR setting as it enables the use of a threshold on the distance to perform known-unknown separation, exactly as we did in HyMOS:

$$\hat{y}^{(t)} = \begin{cases} \underset{y^{(s)}}{\text{argmin}}\,(d_a(h_{y^{(s)}}, \boldsymbol{z}^{(t)})) & \text{if } \min_{y^{(s)}}(d_a(\mathbf{h}_{y^{(s)}}, \mathbf{z}^{(t)})) < \tau \\ \text{unknown} & \text{otherwise} \end{cases} \tag{5.8}$$

The value of the threshold τ is one of the most important choice for an open-world approach (see Table I in [41]). In our case, we can take advantage of the feature space statistics to define the threshold:

$$\tau = [sigmoid\,(\lambda - b) + 1] \cdot \phi \cdot (\ln(\lambda) + 1) \tag{5.9}$$

Differently from HyMOS' case, here we have the constraint imposed through Eq. (5.7) which makes the result of the logarithm always positive. We introduce the hyperparameter b in the first term of the formula (between squared brackets) to provide a way to control the importance of unknown data w.r.t. known one. This hyperparameter allows us to obtain a good known-unknown balancing in all the testbeds considered in the following experimental analysis. We summarize the meaning of our threshold definition in Figure 5.5.

5.4.4 Experimental protocol

We assess the performance of COW by mainly relying on the benchmark proposed in [41], but we also extend it in order to include more recent literature and additional datasets that are relevant to the studied problem and to its possible real-world applications.

Datasets

We focus on five datasets, all of them representing collections of daily-life objects (spanning from fruits and vegetables to tools and containers) captured under very different acquisition conditions.

RGB-D Object dataset (ROD) [76] is one of the most used RGB-D datasets in the literature studying object categorization for robotics. It represents objects placed on a table and captured with different viewpoints. Data collection has been performed in a strictly controlled environment without any source of noise *i.e.* without clutter, and with a fixed illumination and background.

Synthetic ROD (synROD) [97] is a synthetic version of ROD, designed to analyze the synthetic-to-real domain shift problem in a robotic context. It has been defined using publicly available 3D models rendered through a ray-tracing engine in Blender to simulate photorealistic lighting.

Autonomous Robot Indoor Dataset (ARID) [96] is a challenging dataset of pictures of objects captured in a cluttered environment: the same objects appear with different backgrounds, scales, views, lighting conditions, and levels of occlusions. The purpose of this dataset is to evaluate the robustness of a recognition model when dealing with difficult but realistic scenarios.

Continual Open Set Domain Adaptation for Home Robot (COSDA-HR) [69] is a dataset composed of a source domain with pictures of hand-held objects placed in front of a uniform background and a target domain with pictures of the same and other objects captured in various natural locations in a home environment.

Continuous Object Recognition 50 (CORe50) is a collection of photos of domestic objects, captured while being held by the operator in 11 distinct sessions (8 indoor and 3 outdoor).

For the first three datasets we follow the experimental protocol proposed in [41]: among the 51 object categories that they share, we randomly choose 26 to act as *known* while the rest are kept as *unknown*. The incremental protocol is composed of an initial set of 11 base categories and three incremental steps each one including additional five categories. For COSDA-HR instead, we follow [69]: its 160 *known* categories are learned incrementally 10 at a time for a total of 16 tasks. There is also a single *unknown* category composed of a heterogeneous set of objects. In the case of CORe50, which was designed to perform instance classification on 50 objects, we consider 10 of them in the first learning episode and add 5 in each of the subsequent three, keeping the last 25 as *unknown*. We consider the *indoor* $\rightarrow$ *outdoor* domain shift.

In order to better assess the performances of the methods, for all the experiments, we consider five different random class orders and we report the obtained average.

Metrics

For the evaluation we select the metrics employed in [41], which are described in Sec. A.5.2.

Competitors

In [41], Fontanel et al. analyze the performance on the CD-OWR task of a number of state-of-the-art OWR methods, enhanced with single-source DG approaches to deal with the domain-shift.

We thus adopt the same approach to build comparisons and assess the performance of COW, but besides the methods originally considered in [41], we also include an additional CIL state-of-the-art method and an additional DG state-of-the-art method. The methods considered are thus:

- **NNO** [6] a non-parametric approach that exploits the Nearest-Class Mean (NCM) algorithm [107] to compute the class centroids with features extracted through a pre-trained deep architecture;
- **DeepNNO** [105], an improved version of NNO in which the feature extractor is trained end-to-end and the rejection threshold is not fixed but updated during training,
- **B-DOC** [40], an algorithm that includes two clustering constraints in the optimization process and proposes a class-specific rejection threshold;
- **SS-IL** [2], a state-of-the-art CIL method that uses separate softmax output layers combined with task-wise knowledge distillation to mitigate the bias toward the new classes. Considering that SS-IL has not been designed for OWR, but for CIL, it does not include a known-unknown separation procedure. In order to adapt it to our scenario we thus exploit the standard Maximum Softmax Probability approach [50] for defining a *normality* score and we define a threshold using the logic proposed in [77].

For what concerns the DG literature we consider three methods used in [41]: **RSDA** [161] a data augmentation-based technique, **RR** [97] a self-supervised-based technique, and **SC** [56] a regularization-based strategy. Moreover, we include the more recent DG approach **SagNet** [117] that disentangles the sample content and style to let the network focus more on the first than on the second.

Implementation Details

We implement COW reproducing the protocol adopted for all our competitors: we use a ResNet18 backbone trained from scratch using images of size 64×64. When learning

Table 5.6: Results (%) averaged over five random class orders.

OWR/CIL	DG	ROD → ARID			synROD → ARID			synROD → ROD			COSDA-HR			CORe50		
		Acc-WR	Acc	OWR-H	Acc-WR	Acc	OWR-H	Acc-WR	Acc	OWR-H	Acc-WR	Acc	OWR-H	Acc-WR	Acc	OWR-H
NNO [6]	-	18.4	3.1	5.9	16.2	7.8	13.7	21.3	13.3	21.1	8.2	3.7	7.0	15.0	0.6	1.2
DeepNNO [105]	-	21.3	7.3	13.4	15.9	5.4	10.0	24.4	9.6	17.0	15.1	8.2	12.8	17.0	4.5	8.1
B-DOC [40]	-	22.3	10.0	17.5	16.5	2.2	4.3	27.6	5.2	9.9	13.2	0.7	1.3	15.7	2.2	3.8
SS-IL [2]	-	17.6	14.7	16.4	21.3	9.6	16.1	29.3	16.9	22.9	8.4	5.0	6.2	23.2	16.8	18.6
NNO [6]	+ RR [97]	27.1	13.6	21.7	15.8	7.2	12.5	25.9	17.1	23.8	8.2	3.2	6.5	14.5	0.4	0.9
DeepNNO [105]	+ RR [97]	33.5	16.0	25.8	14.2	4.9	9.3	34.1	15.4	25.2	13.8	6.9	11.0	17.5	5.0	8.9
B-DOC [40]	+ RR [97]	32.2	11.7	20.4	15.7	2.2	4.3	35.9	9.7	17.3	12.3	0.5	1.0	19.5	4.5	7.1
SS-IL [2]	+ RR [97]	17.3	13.0	17.9	19.7	6.6	11.6	30.9	16.2	23.7	7.8	1.4	2.6	20.2	9.7	12.8
NNO [6]	+ SC [56]	14.1	9.8	15.5	16.0	11.6	16.9	21.9	18.8	21.2	6.4	4.6	7.2	13.3	2.9	5.3
DeepNNO [105]	+ SC [56]	20.9	15.9	22.0	15.5	8.4	14.6	25.9	17.0	25.3	15.3	11.7	15.5	18.2	8.8	13.6
B-DOC [40]	+ SC [56]	19.6	13.1	20.4	16.5	10.0	16.1	26.7	18.0	23.2	13.0	1.9	3.3	17.1	4.7	6.8
SS-IL [2]	+ SC [56]	15.2	12.9	14.7	19.0	7.9	13.2	26.8	14.6	21.0	9.0	5.5	6.6	19.3	14.7	16.0
NNO [6]	+ RSDA [161]	25.0	12.8	20.7	16.3	8.6	14.4	26.7	18.4	24.5	8.9	2.1	3.9	22.1	13.1	16.1
DeepNNO [105]	+ RSDA [161]	33.3	14.9	24.6	15.3	4.2	8.0	34.2	14.0	23.5	18.4	10.9	17.1	38.0	20.8	30.7
B-DOC [40]	+ RSDA [161]	31.9	12.2	21.1	16.3	2.5	4.9	37.9	10.8	19.1	18.2	0.6	1.0	**41.4**	9.8	15.9
SS-IL [2]	+ RSDA [161]	29.9	**24.4**	24.1	20.3	7.6	12.8	**38.7**	23.6	30.3	17.8	8.3	12.6	38.1	**25.9**	30.6
NNO [6]	+ SagNet [117]	19.1	3.9	7.4	15.2	7.3	12.7	20.3	12.4	19.4	8.1	3.2	6.4	15.9	2.5	4.7
DeepNNO [105]	+ SagNet [117]	22.5	8.7	15.5	13.7	4.7	8.8	17.9	7.1	12.8	8.5	3.9	7.2	19.3	7.6	12.4
B-DOC [40]	+ SagNet [117]	23.7	10.7	18.2	18.2	4.6	8.5	28.9	9.1	16.1	11.3	0.5	1.1	17.3	3.6	5.9
SS-IL [2]	+ SagNet [117]	24.9	19.4	21.8	20.9	8.9	14.9	32.8	17.7	24.5	8.3	3.7	6.1	27.3	17.6	23.5
COW		**34.0**	18.8	**28.6**	**29.8**	**21.3**	**28.1**	34.1	**24.0**	**30.7**	**20.1**	**16.2**	**21.4**	33.9	23.9	**32.9**

a new task we keep a fixed-size memory buffer to store $M = 2000$ randomly selected samples of the classes of previous tasks. We train each task until our stop training condition (Eq. 5.7) is matched. COW has only two hyperparameters, ϵ and b, and is robust to their value as discussed in Sec. 5.4.5.

5.4.5 Experimental analysis

In our experiments, we use the *source* → *target* notation to identify the various scenarios. The results reported in the tables and plots show how existing OWR and CIL solutions are far from solving the task of cross-domain open-world recognition even if enhanced with DG approaches to bridge the domain gap. On the contrary, COW, using a single loss function and without any dedicated cross-domain module, is able to mitigate the domain shift and outperform the current state-of-the-art.

Main results

We start analyzing the performance of vanilla state-of-the-art OWR and CIL approaches on the CD-OWR task without the help of DG methods, reporting the results in the upper part of Table 5.6. As already reported by Fontanel *et al.* in [41] these methods perform poorly in a cross-domain scenario. The second block of the table presents the results obtained by combining the same approaches with single-source DG ones. We reproduced the experiments originally included in [41], enriching them with the more recent methods SS-IL and SagNet.

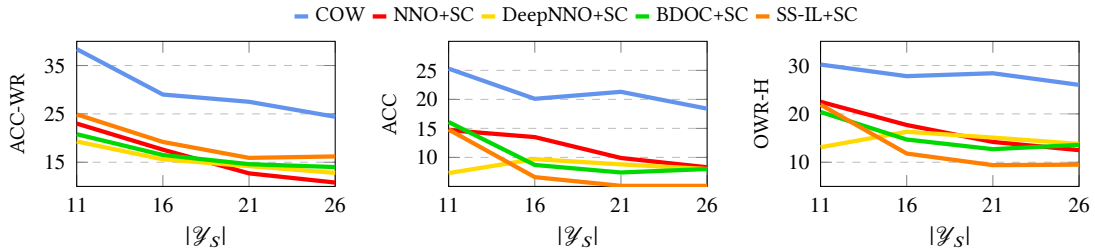


Figure 5.6: Performance analysis at subsequent learning episodes for synROD → ARID. The number of known classes $|\mathcal{Y}_S|$ increases and the plots show how COW maintains a consistent gain over all the competitors.

We discuss the results focusing on the OWR-H metric that was shown to be the most appropriate to evaluate open-set approaches [41], as was the case fot HOS in open-set domain adaptation [14]. For what concerns the ROD → ARID and synROD → ROD domain shifts, as well as the CORe50 dataset, we can generally see an improvement after adding each one of the DG approaches. In general, the non-negligible average improvement gained when exploiting a DG strategy confirms the generalization failure of the original OWR/CIL approaches. Moreover, among the considered DG strategies the one that most often produces the highest results is the data augmentation-based approach RSDA, which confirms the great advantage that a strong data augmentation can provide in knowledge generalization, as largely discussed in Chapter 4. In some edge cases, however, even this approach is not enough. If we focus on the synROD → ARID shift, and on the COSDA-HR dataset, we see that all the DG strategies do not seem to provide a significant and consistent improvement. In both these cases, the domain shift is quite severe since it includes a (realistic) target domain with images recorded in a cluttered environment, very different from the neat (and possibly synthetic) training set. The considered DG strategies are clearly not suited to reduce such a large domain gap.

Anyway, the table shows that COW obtains the best results over all the experiments, proving its effectiveness.

Domain Generalization through Contrastive Learning

Contrastive learning relies on data augmentation to create augmented views of the training samples and learn invariance to those augmentations. The employed augmentation techniques are more or less the same used in RSDA [161], with the addition of random resized crop (RC). In order to assess whether the good results of COW originate mainly from this augmentation pipeline, or from the specific way in which it is used by the contrastive loss, we propose an analysis in which we provide other baselines with the additional RC transformation. In Table 5.7 we consider the synROD → ARID domain shift. We compare against our best competitor for this shift, which is

Table 5.7: Contrastive learning vs Data augmentation.

OWR/CIL	**DG**	**synROD → ARID**		
		Acc-WR	Acc	OWR-H
NNO [6]	+ SC [56] + RC	16.5	13.8	14.6
NNO [6]	+ RSDA [161] + RC	17.3	12.5	14.5
DeepNNO [105]		20.4	10.7	17.7
B-DOC [40]		23.8	13.3	20.1
SS-IL [2]		27.0	11.2	17.9
COW		**29.8**	**21.3**	**28.1**

NNO+SC, but also against the methods that already use the strong augmentations of RSDA. We observe that the performance of the considered competitors does not always increase with the considered addition, highlighting that this augmentation can hurt the performance of methods that are not designed to manage it. Nevertheless, even when the performance of the competitors increases, COW still keeps the best results. This evidences that data augmentation is not enough by itself to enable generalization and thus that the contrastive logic plays a fundamental role in this sense.

Incremental learning performance

The performance over multiple incremental steps is a fundamental element to consider when comparing incremental learning methods. We analyze it by reporting the scores for subsequent learning episodes in Figure 5.6 on the synROD → ARID domain shift. On this specific shift, the Self Challenging (SC) approach is the DG method providing the higher mean improvement to COW's competitors, thus we focus on it for the comparison. For all three metrics, we can see that COW keeps a large performance gap over the other approaches for all the incremental steps. After a natural decrease in accuracy following the first task, in the subsequent learning episodes COW is able to maintain a quite stable ACC and OWR-H performance showing a great ability to balance the accuracy on known and unknown samples. We remark here that COW does not include any sophisticated incremental learning technique, which means that its excellent results come from the judicious combination of contrastive learning with a fixed-size replay buffer and class-balanced sampling for mini-batches.

Sensitivity Analysis

COW exploits two hyperparameters, ϵ and b, to balance the training and the inference process in order to obtain a well-structured feature space that supports known-unknown separation. Their influence on the model's performance is different and depends on the value of λ, which in turn depends on the statistics of the training dataset. We analyze the situation considering two different shifts representing two extreme

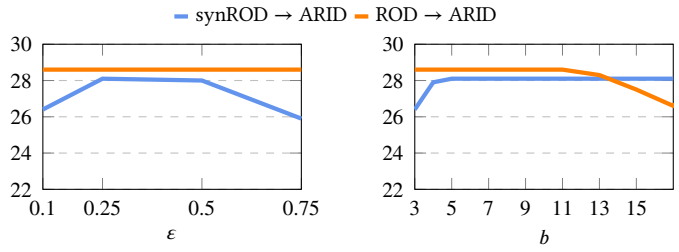


Figure 5.7: OWR-H results when varying ε and b.

cases: for synROD → ARID we have $\lambda \approx 1 + \varepsilon$ (a situation similar to Fig. 5.5 left), while for ROD → ARID we have $\lambda \gg 1$ (Fig. 5.5 right). We report the results of this analysis in Fig. 5.7.

The value of ε controls the minimum margin between known class clusters imposed through the stop training criterion of Eq. (5.7). As a consequence, a larger ε pushes the training towards a larger margin by increasing clusters compactness and separation. While this condition may seem always desirable, a too-high value may in practice lead to overfitting, without considering the very long training times necessary to reach the stop training criterion. As the plot on the left shows, the value of ε does not influence the performance on ROD → ARID as for this shift the margin is naturally quite high.

The hyperparameter b tunes our known-unknown separation threshold τ (see Eq. (5.9)), and it allows us to find a good balance between known and unknown accuracy.

The main outcome of this analysis is the proof that the results obtained by COW are stable and high ($\geq 26\%$) for a reasonable range of hyperparameters' values, and always outperform the best competitor (e.g. for synROD → ARID, NNO + SC obtains 16.9, for ROD → ARID, DeepNNO + RR obtains 25.8).

5.5 Conclusions

In this chapter, we have studied two different open-world learning settings, both characterized by a multitude of challenges **including a semantic and a visual distribution shift**. The concurrent occurrence of these two types of distribution shift makes it even more difficult to obtain robust performance after deployment, considering that this means ignoring completely one shift type while detecting all the occurrences of the other. The solutions that we have proposed are based on the idea that **the correct choice in terms of learning objective** allows us to simplify significantly the management of these complex problems. Indeed, the adoption of supervised contrastive as the main learning function enables overcoming some of the problems of the standard cross-entropy one, such as the **prediction overconfidence** and the **attention to local features**, ultimately leading to a reduction of the supervision collapse issue. Indeed, the explicit invariance to a bunch of semantic-preserving transformations obtained by contrastive learning forces the network to focus on global features, while the obtained strongly structured feature space enables the definition of sensible strategies for classification and known-unknown separation. The significant impact of this structure on OOD performance has been later noted also in several other works [112, 85].

This first analysis of the advantages that can be obtained by modifying our approach to representation learning opens the way for a broader discussion on this topic. Up to this point, we have always focused on strategies that learn representations directly on the task on which they are applied, by training from scratch or fine-tuning a model on the data of the task at hand. This approach may be suboptimal w.r.t. representation learning paradigms with wider scopes applied on huge data collections at scale. These are the topics under analysis in the next chapter, where for the first time we analyze the impact of foundation models on distribution shift problems. For what concerns these models, it is interesting to note that, despite a significant scale difference, foundation models designed for the Computer Vision field have in common with HyMOS and COW the adoption of contrastive-based learning objectives [122, 129], demonstrating one more time the superiority of this learning paradigm in terms of knowledge generalization.

Chapter 6

Out-Of-Distribution detection beyond fine-tuning

The standard practice for tackling **Out-Of-Distribution detection** relies on full training, or at least a fine-tuning of a pre-trained model, on the in-distribution data of the task at hand. This approach has a number of disadvantages, the first of which is the poor out-of-distribution representation capabilities it provides, caused by *supervision collapse* and *forgetting* of any general pre-trained knowledge. In spite of this, this approach has certainly been the most suitable to deal with OOD detection for a long time, but we argue that the situation has recently started to change. Indeed, the recent evolutions in terms of neural network architectures, distributed and large-scale training, and availability of very large data collections have begun to make competitive an alternative approach that in the past would have only guaranteed poor results: the direct use of pre-trained knowledge to tackle downstream tasks without model fine-tuning. The purpose of this chapter is to deeply explore this alternative approach. We start by proposing a relational reasoning-based representation learning paradigm, which acts as a proof-of-concept of the possibility of performing Out-Of-Distribution detection without fine-tuning. Indeed, the semantic similarity measure it provides has wide applicability, and thanks to a thorough analysis of alternative learning objectives, it also shows great generalization capabilities. We then frame this approach in the bigger picture of a large-scale experimental comparison between fine-tuning-free and fine-tuning-based OOD detection strategies, for which we consider a wide number of pre-training solutions in order to assess their applicability to the studied problem. In this context, for the first time, we consider also ***foundation models***, in order to assess the impact that these novel general-purpose feature extractors can have on a whole research field.

Part of the work described in this chapter has been previously published in two papers:

- [18] F. Cappio Borlino, S. Bucci, and T. Tommasi
  Semantic Novelty Detection via Relational Reasoning
  European Conference on Computer Vision, ECCV 2022

- [101] L. L. Lu, G. D'Ascenzi, F. Cappio Borlino, and T. Tommasi
  Large Class Separation is not what you need for Relational Reasoning-based OOD Detection
  International Conference on Image Analysis and Processing, ICIAP 2023

- [20] L. L. Lu, F. Cappio Borlino, and T. Tommasi
  Foundation Models and Fine-Tuning: A Benchmark for Out Of Distribution Detection
  IEEE Access, 2024

6.1 Out-Of-Distribution detection with pre-trained representations

Deep learning-based solutions are increasingly been chosen for integration in real-world Computer Vision applications, thanks to the excellent performance they provide w.r.t. traditional algorithms. As we have seen, this deployment often leads to the emergence not only of visual distribution shifts but also of semantic ones. Indeed, any deployment in an open-world scenario may lead an autonomous agent to face samples belonging to semantic categories it doesn't know and that it shouldn't confuse with known classes. This situation is associated with safety risks whose severity depends on the application, but that can reach *critical* levels in some specific cases, for example for autonomous driving. For this reason, it has started attracting the interest of researchers, who have defined a number of research settings that take this scenario into account (see Sec. 2.3.1). Among these settings, the one that formulates and studies the problem in the most general way is **Out-Of-Distribution detection**.

The research in this field has, very recently, started recording a **paradigm shift**, in the wake of what is happening in the CV world in general with the rise of vision-based *foundation models* [129, 122]. These are deep models trained *at scale* on huge amounts of data, able to provide all-purpose representations [9]. Despite being generally based on standard self-supervised learning strategies, simply applied at a scale not previously possible, foundation models learn *task-agnostic* representations, which can be used "as they are" on downstream tasks, often outperforming the results obtained by task-specific models [122]. After conquering the Natural Language Processing field, foundation models have recently begun reaching the CV one. One of the first consequences of their appearance is that the traditional strategy of using a model pre-trained on ImageNet1k [34] as a starting point for doing *transfer learning* through fine-tuning on downstream data, is becoming obsolete, with more and more research papers focusing on developing strategies to exploit the knowledge encoded by foundation models [191, 188, 111]. Indeed, a significant challenge when using a large-scale pre-trained network for a downstream task is to fully exploit the knowledge it encodes, without damaging it. The traditional full network fine-tuning, which is designed to obtain good performance on the downstream task training data distribution, leads the model to overfit that same distribution while forgetting all the knowledge not necessary for the fine-tuning task. The obvious consequence is that the model's representation capabilities on OOD data fall considerably [74]. Of course, if the pre-trained model is not powerful enough to provide acceptable performance on In-Distribution (ID) data, model fine-tuning may still be the best solution to exploit its knowledge and this is often the case when relying on traditional ImageNet1k classification pre-trainings. However, with the rise of foundation models, this forced trade-off between ID and OOD performance may need a *renegotiation*, especially in those cases in which OOD performance is particularly important, as it happens for the OOD detection task.

Indeed, *traditional* solutions designed for this task **explicitly require** complete training, or at least fine-tuning on the downstream task ID data (we call them *fine-tuning-based* strategies). For a long time, this training step has been considered necessary, in order for the model to automatically learn, by embedding it into its weights, the train data distribution, and later be able to detect samples deviating from it. Recently, however, there have been some efforts to **develop a more general approach**, by exploiting pre-trained models directly for comparing test data with a *support* set of samples representing the normality, hence **completely discarding** the fine-tuning step [111, 164, 18, 101] (thus the name *fine-tuning-free* approaches). This strategy is flexible and computationally efficient as it allows using the same pre-trained model for different OOD detection tasks. Moreover, the same *generic* feature extractor is used to represent both ID and OOD data, which means that its representation capabilities are not restricted to the ID classes.

The simultaneous emergence of foundation models, with their excellent general-purpose representation capabilities, and the appearance of the first fine-tuning-free OOD detection algorithms, raises the question of what real performance can be expected when these two elements are combined.

In this chapter, we aim to answer this question. We will start by presenting a representation learning strategy specifically designed to build a model that supports OOD detection without fine-tuning. We later show how distance-based evaluation approaches can be applied on top of any feature extractor, thus enabling the exploitation of both traditional pre-trained and foundation models for OOD detection. We then proceed by performing a comparison between all these solutions and approaches coming from the literature, with the aim of building a comprehensive picture of the state-of-the-art in fine-tuning-free OOD detection.

In detail, the key **contributions** of this chapter are:

- the presentation of a novel **representation learning** paradigm, based on *relational reasoning*, which aims at training a model to provide a measure of semantic similarity. This measure can be used to distinguish between semantic classes and thus naturally supports an application to OOD detection without needing fine-tuning. Through an analysis of alternative solutions for the learning objective of this approach we also highlight how a characteristic of common loss functions, usually deemed a quality, can on the contrary represent an obstacle to features re-use;

- the design of a **novel comprehensive benchmark** for OOD detection on 2D data, which includes five *intra-domain* settings designed to evaluate the semantic novelty detection ability on a wide variety of categorization tasks (from object-centric images to aerial ones, from textures classification to fine-grained car recognition and scene categorization), and a set of *cross-domain* scenarios, designed to evaluate the models' performance in a realistic deployment setting in which semantic and domain shift appear together;

- a **large scale comparison**, carried out on the novel benchmark, between **fine-tuning-free** approaches and traditional **fine-tuning-based** ones, which highlights the advantages and disadvantages of the two strategies.

Our comparison involves both algorithms specifically designed to tackle the OOD detection task, and pre-trained models that on the contrary have been designed with other goals in mind. These models are thus considered mainly as general-purpose *feature extractors*, and it's on the features that they provide that OOD detection is performed. Our analysis aims to discover which of these pre-trainings provides the most suitable representations to support the considered task.
Note: We generally use the term "pre-training" meaning "training *from scratch* with a representation learning objective". Hence the "pre-training objective" and the "pre-training dataset" represent precisely the adopted learning objective and the dataset on which it is applied, while the "pre-trained model" is the deep network output of this process, in most cases used as a *feature extractor*, *i.e.* as a module that allows to extract representations from images.

6.1.1 Related works

Representation learning

It is probably the most important difference between the classic *shallow* and the modern *deep* machine learning approaches. Indeed, the former exploited handcrafted features, manually designed by experts to support various CV applications, while the latter is based on the automatic learning of representations, obtained as part of the process of training a model on a specific task. This deep networks' ability has enabled the solution of more complex problems, but at the same time has paved the way for the emergence of phenomena like *supervision collapse* [36].

In the beginning, representation learning was mainly performed through supervised learning approaches, generally involving quite large labeled datasets such as ImageNet1k [34]. During the last years, significant attention has also been devoted to learning representations from unsupervised datasets, through the development of self-supervised representation learning paradigms [44, 121] with a particular focus on contrastive-based approaches [49, 26, 24]. In all cases, however, the usual approach to exploit the learned representations for a downstream application was through a transfer learning procedure involving a fine-tuning on annotated data of this downstream task. In most cases, this could not be avoided, as the representations extracted by pre-trained models were not powerful enough to support their direct use. The situation has started to improve with the introduction of larger pre-training solutions, which became necessary with the continual growth in size of deep networks. Part of this process involved starting to use larger datasets, such as ImageNet21k [34], and the switch from convolutional networks to vision transformers [37]. In other cases, the representation learning paradigm has been modified explicitly to support transfer learning [70].

More recently, the improvements in terms of contrastive learning and scalability of deep networks training have enabled the presentation of large vision-language pre-training solutions such as CLIP [129] and ALIGN [60], but also of purely vision-based self-supervised pre-trainings as DINO [24] and DINOv2 [122]. The huge scale of some of these pre-training solutions enables the obtained models to provide high-quality general-purpose representations that can be used for downstream tasks directly. For this reason, they can be seen as the first foundation models [9] for Computer Vision.

Relational reasoning

It is a hallmark of human intelligence and it can be seen as the ability to quantify the relationship between a set of objects. Even before the appearance of large scale vision-language models such as CLIP, this paradigm has attracted attention for the combination of vision and language for scene description [63, 141]. However, it has also been applied in different fields, as in few-shot learning [151] and vision-only self-supervised learning [125]. In particular, this last paper showed that relational reasoning can represent an alternative paradigm to the contrastive approach for effectively learning powerful representations in a self-supervised fashion. It should be noticed that, even if both these paradigms base their learning on building comparisons between pairs of samples, there is a fundamental difference between them: contrastive learning aims at building a feature space in which single points represent individual samples, while relational reasoning obtains representations in which each point is a sample pair.

Out-Of-Distribution detection without fine-tuning

Most of the literature studying OOD detection focuses on approaches requiring complete training or at least fine-tuning of a neural network on the downstream task in-distribution data; this is the case for the papers presented in Sec. 2.3.1. Given the high cost in terms of data and computational resources of performing this fine-tuning step, which adds to the aforementioned disadvantages in terms of forgetting any broader pre-trained knowledge, some recent works have started proposing strategies to avoid it completely. In particular, besides the relational reasoning based approach that we will describe in more detail later and that was presented in [18], significant efforts in this sense have been directed at the development of strategies exploiting large vision-language pre-trainings like CLIP [129]. In particular, MCM [111] proposed to use *concept prototypes*, created using ID class names and CLIP's text encoder, as a summary of task *normality*. By adopting this approach, the computation of a *normality* score for a test sample is as simple as extracting its visual embedding through CLIP's image encoder and computing its similarity with the nearest known class prototype. This strategy is in practice a naïf application of CLIP's zero-shot capabilities to the OOD detection task. More sophisticated approaches have been proposed in [164] and [110],

which study parameter-efficient algorithms to adapt CLIP to downstream OOD detection tasks. While these methods need an adaptation phase which may require some gradient descent steps for updating a set of parameters, this is significantly more efficient than performing a full network fine-tuning and it does not involve any risk of knowledge forgetting. Besides methods specifically designed to perform OOD detection without fine-tuning, in [101], Lu *et al.* pointed out that some distance-based OOD detection methods presented initially to be used on top of a fine-tuned model [149, 79] may equally be applied on top of pre-trained models, sometimes even obtaining surprisingly good results.

6.1.2 Problem formalization

In this chapter, we focus on the Out-Of-Distribution detection problem as presented and formalized in Sec. 2.3.1.
In the context of *fine-tuning-free* OOD detection, the *support* set $\mathcal{S}$, even if not used to perform an actual training, clearly still represents and summarizes the *normality* of a given task, which detectors should be able to exploit in order to identify test samples that deviate from it. The strategy of use of the *support* set is an important part of the definition and design of an OOD detector.

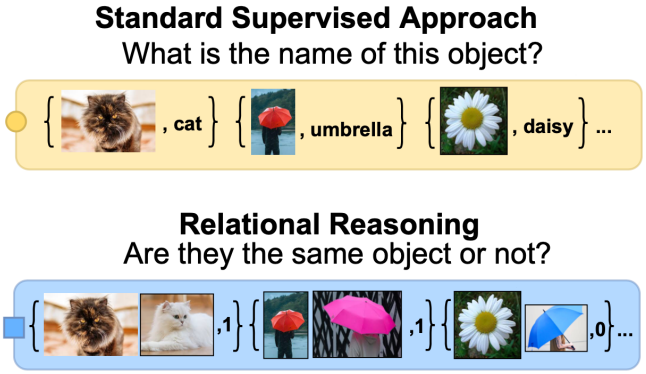


Figure 6.1: A visual comparison of relational reasoning-based representation learning w.r.t. the traditional supervised categorization approach

6.2 Relational reasoning supports fine-tuning-free OOD detection

6.2.1 Representation learning via Relational Reasoning

Representation learning is the task of learning how to represent complex data structures in order to support the solution of a specific problem. The *innate* representation learning capabilities of deep neural networks are considered one of their main advantages w.r.t. traditional shallow machine learning. This ability is *innate* as representation learning is obtained as a *byproduct* of the training of the network on a specific task. For example, in the most common scenario, a deep network is trained for categorization in a supervised manner, which means that it is trained to associate each training sample to a specific semantic category, learning to answer the question *what is the name of this object?* One of the consequences of this approach is that the learned representations are not generic, but are rather task-specific, which means that they are perfectly suited to solve the task on which the model has been trained, but may not generalize to other tasks.

The traditional approach used for Out-Of-Distribution detection involves training a model from scratch, or fine-tuning a large pre-trained model, on the data of the task at hand in order to push the model to embed in its weights the concept of *normality* for the considered task. In many cases, for example for the *discriminative* approaches (see Sec 2.3.1), this training is performed with a categorization learning objective applied to known data. As a consequence, the learned representations are perfectly suited to categorize nominal samples but may be unsuited to correctly represent OOD ones [74].

We propose thus to apply a different paradigm for the choice of the representation learning strategy that should be adopted when the final goal is to perform OOD detection. In particular, we choose a relational reasoning learning objective in which a

model is trained to recognize if two samples in a pair come from the same semantic category, learning to answer the question *are they the same object or not?* We summarize the difference between this approach and the traditional supervised categorization one in Figure 6.1.

The output provided by these two representation learning strategies is radically different. When considering a categorization learning objective we can imagine the learned feature space as a space where each input sample is represented as a single point, with its position identified through a vector. Samples of the same class are grouped in a cluster and the categorization task is performed by identifying decision boundaries that separate clusters referring to different semantic classes.

On the other hand, if we consider a relational reasoning-based learning objective, the final feature space hosts points that represent input sample pairs. The position of a point, identified through a vector, indicates whether the two samples composing the pair belong to the same semantic class or not. This space can be seen as the output of a training performed on a binary categorization task, where the two classes summarize the concepts of *same* category and *different* categories. A model trained on this objective is thus capable of providing a measure of semantic similarity between pairs of samples.

6.2.2 Relational reasoning applied to OOD detection

The main advantage of the adoption of a relational reasoning-based learning objective is that the learned concepts are much more generic than the ones learned through a categorization approach, provided, of course, that the training is performed on a dataset large enough to support generalization. Indeed, the feature space obtained as the output of representation learning is suited only to represent samples belonging to the categories encountered during training, and in this case, *same* and *different* are generic concepts that can be applied even when the members of a pair come from previously unseen classes.

We propose to exploit this genericness to perform OOD detection. With this goal in mind, we exploit the ability of our model to measure semantic similarity to compare the *support* set samples, which represent the normality, with the test ones. A test sample normality score can thus be estimated by using its maximum similarity with a representative of *known* data. If the *support* set contains a large number of samples for each known class, a *prototype* for each class can be chosen in order to reduce the number of comparisons that need to be performed at inference time.

We dub the overall OOD detection algorithm that we obtain **ReSeND**: Relational reasoning for Semantic Novelty Detection.

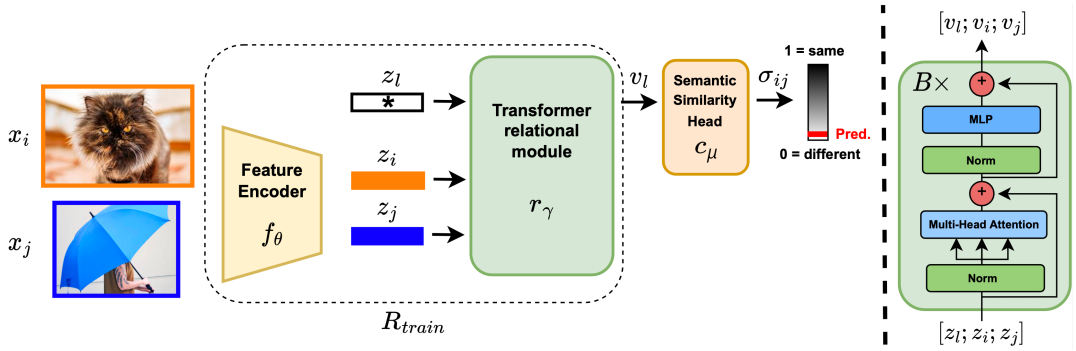


Figure 6.2: Schematic illustration of our relational reasoning network architecture

6.2.3 Design of a network for relational reasoning

A network designed for relational reasoning must be able to process pairs of input samples jointly. We consider a model composed of three main modules (see Fig. 6.2):

- a feature encoder/extractor f_θ. This module takes in input individual samples and provides as output vectorial representations for them: $\boldsymbol{z} = f_\theta(\boldsymbol{x})$. It can be implemented by adopting a standard CNN-based encoder, in our case we use the backbone of a ResNet18 pre-trained on ImageNet1k;
- a relational module r_γ. This module is tasked to aggregate the representations extracted for a pair of input samples and provide in output a single representation for the pair;
- a semantic similarity head c_μ, which takes in input the output of the relational module and provides as output a single scalar value representing the measure of semantic similarity $\sigma_{ij} = c_\mu(r_\gamma(\boldsymbol{z}_i, \boldsymbol{z}_j))$. If we restrict $\sigma_{ij} \in [0,1]$ we can interpret this value as the probability that the two input samples belong to the same semantic category.

The tuple $\{\theta, \gamma, \mu\}$ collects all the trainable parameters of our network.

Relational module It is the most important module of our network, as it processes a pair of samples to provide in output a single representation aggregating the information of the pair. It can be implemented in various ways with the simplest one being a Multi-Layer Perceptron (MLP) taking in input a concatenation of the features of the pair. We propose a slightly more sophisticated solution that builds on the transformer [157] architecture, because of its well-known capability of comparing multiple inputs and its natural permutation invariance. In particular, our relational module consists of B identical blocks, each one composed of a Multi-Head Self-Attention (MSA) and a MLP, both preceded by Layer-Norm (LN) modules and bypassed by residual skip connections as shown in the right part of Fig. 6.2. We provide in input to the first block a sequence

$[\boldsymbol{z}_l, \boldsymbol{z}_i, \boldsymbol{z}_j]$, where $\boldsymbol{z}_l$ is a *learnable* token, and get as output from the last block a similar sequence $[\boldsymbol{v}_l, \boldsymbol{v}_i, \boldsymbol{v}_j]$. In this architecture, each input image $\boldsymbol{x}$, encoded as $\boldsymbol{z}$ by the feature extractor, represents a single input token for the transformer, as done in [28]. From the output sequence, we select only the token $\boldsymbol{v}_l$ as output of the whole relational module which is later passed to the semantic similarity head c_μ. It should be noted that the output of a good relational module should be invariant to the order of appearance of the two samples composing a pair so that the final similarity measure provided by the overall network is symmetric. Specifically, to obtain this characteristic we avoid the inclusion in our network of any kind of positional encoding, which is on the contrary a standard practice when using transformers to process sequences in which the order of token matters.

Inference procedure One of the main advantages of the proposed architecture is that, besides providing as main output a semantic similarity measure for a pair of samples, it naturally also provides as intermediate output individual representations for the input samples, which can be extracted by the feature encoder f_θ. This ability can be exploited at inference time to preprocess the *support* set and extract a compact encoding of the normality. This is done by obtaining the feature representation of all *support* samples and then performing a per-class feature average to compute class prototypes. When performing inference on a test sample $\boldsymbol{x}_t$ its representation $\boldsymbol{z}_t$ is paired with all known class prototypes in turn, and the pairs obtained are processed through the relational module and the semantic similarity head. The final normality score for the test sample is simply the maximum value among the similarity scores obtained in this way.

6.2.4 Training of a relational reasoning network

The relational reasoning task can be framed both as a binary classification and a regression problem. In the former case, we can see the model as a classifier over the {*same*, *different*} class set, in the latter as a regressor for a continuous semantic similarity measure. From a conceptual point of view, this framing is not particularly important, as what we want at the end of the day is just a module able to estimate the semantic similarity of sample pairs. A much more practical problem is represented by the choice of the learning objective, whether this comes from the field of regression or classification. Indeed, our main interest here is to maximize the generalization ability of the final model as we aim to apply it to a possibly wide range of OOD detection tasks without fine-tuning it. The choice of a learning objective is particularly sensible as this element has the power to shape the structure of the final feature space [71], thus greatly impacting the generalizability of the trained model.

We thus analyze a number of possible alternative loss functions and study how they impact on the learned feature space.

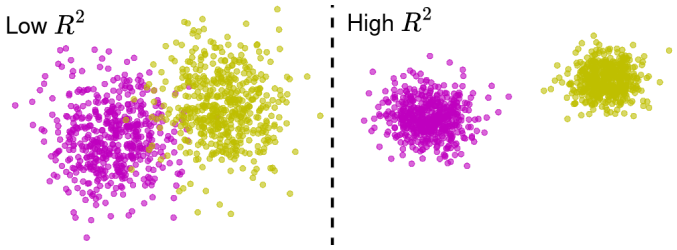


Figure 6.3: Toy example of feature space structures characterized by high or low R^2 index

Class Compactness and Separation As we have already pointed out, the final feature space learned by a relational reasoning model is analogous to the feature space learned by a binary classification task training, characterized by the presence of two class clusters, representing the *same* and *different* semantic concepts. In order to analyze the characteristics of such feature space we leverage the R^2 index originally introduced in [71].

This metric is designed to provide a relative measure of the sparsity of the representations composing class clusters in a given embedding space, in particular, it depends on the ratio between the average within-class and global cosine distance between feature vectors:

$$R^2 = 1 - \bar{d}_{within}/\bar{d}_{total} \tag{6.1}$$

Where:

$$\bar{d}_{within} = \sum_{k=1}^{K} \sum_{i=1}^{M_k} \sum_{j=1}^{M_k} \frac{1 - \text{sim}(\boldsymbol{p}_i^{(k)}, \boldsymbol{p}_j^{(k)})}{K M_k^2}$$
$$\bar{d}_{total} = \sum_{h=1}^{K} \sum_{k=1}^{K} \sum_{i=1}^{M_h} \sum_{j=1}^{M_k} \frac{1 - \text{sim}(\boldsymbol{p}_i^{(h)}, \boldsymbol{p}_j^{(k)})}{K^2 M_h M_k}$$

In our case we have only $K = 2$ classes, and the indices $i, j \in \{1, \dots, M_k\}$ are used to iterate over the pairs $p^{(k)}$ of each of these two classes. In this context $\boldsymbol{p}$ is a feature vector representation of a pair, for example the token $\boldsymbol{v}_l$ provided in output by our relational module. The measure adopted for estimating the similarity between two vectors is the cosine similarity:

$$\text{sim}(\boldsymbol{p}_i, \boldsymbol{p}_j) = \frac{\boldsymbol{p}_i^T \boldsymbol{p}_j}{\|\boldsymbol{p}_i\| \|\boldsymbol{p}_j\|} \tag{6.2}$$

A feature space has a high R^2 index if the class clusters are relatively compact and well separated, the R^2 is low when they are less compact or overlapping. We provide in Figure 6.3 a visualization that should clarify the difference between these two cases.

Loss functions for relational reasoning

We consider a number of learning objectives that can be used for binary problems. In our equations we use σ to refer to the score produced as output by the semantic similarity head c_μ for a sample pair $\boldsymbol{p}$. Some of the analyzed loss functions (*e.g.* the Softmax Cross Entropy) need a score to be produced for both classes, in this case we suppose that c_μ has an output of size two, whose values we interpret as classification logits. In all the other cases the single score produced by c_μ can be directly interpreted as a measure of semantic similarity, *i.e.* the score for the class *same*. We use l to indicate the ground truth for the computation of the loss, in other words, this is an indicator of whether the two samples of the pair come from the same semantic class.

Binary Cross-Entropy The Cross-Entropy (CE) loss is one of the most used loss functions when dealing with categorization tasks. Its general formulation is:

$$\mathscr{L}_{\text{CE}} = -\sum_{m=1}^{M}\sum_{k=1}^{K} t_{m,k}\log(\hat{t}_{m,k}) \tag{6.3}$$

where K is the number of classes, while $t_{m,k}$ is the target value and $\hat{t}_{m,k}$ the predicted probability of the class k for the sample m. In particular, for the target value, a *one-hot* encoding is used so that $t_{m,k} = 0$ when the considered category does not correspond with the GT one ($k \neq l_m$).

The **binary version** of this loss function is:

$$\mathscr{L}_{\text{BCE}} = -\sum_{m=1}^{M}\left(t_{m,1}\log(1-\hat{t}_{m,2}) + t_{m,2}\log(\hat{t}_{m,2})\right) \tag{6.4}$$

where $\hat{t}_{m,2}$ is obtained by applying the sigmoid function to the model output:

$$f(\sigma) = \frac{1}{1+e^{-\sigma}}$$

Impact on the class separation: given that this loss is non-zero even for already correctly classified samples the intra-class compactness and inter-class separation keep increasing for the whole training procedure, making the R^2 index value progressively higher.

Softmax Cross-Entropy It is the CE loss version that is used most often for multi-class categorization problems and it is obtained by applying the CE of Eq. 6.3 only after passing the model output through the softmax function:

$$f(\sigma)_k = e^{\sigma_k} / \sum_{c=1}^{C} e^{\sigma_c}$$

Considering the one-hot nature of the labels, it is possible to rewrite the Softmax Cross-Entropy as:

$$\mathscr{L}_{\text{SCE}} = -\sum_{m=1}^{M} \log \frac{e^{\sigma_{m,l_m}}}{\sum_{k=1}^{K} e^{\sigma_{m,k}}} \tag{6.5}$$

where $\sigma_{m,k}$ is the score corresponding to the class k for the sample m and l_m represents its ground truth label. In the binary case we suppose $k, l \in \{1,2\}$.
Impact on the class separation: as was the case for the BCE, this loss takes non-zero values even for correctly classified samples. A number of papers have shown that the final effect of the trend of increasing values for intra-class compactness and inter-class separation leads to badly calibrated classifiers [80, 116, 166], *i.e.* classifiers for which the prediction confidence is not a robust indication of the probability associated to the predicted class. The overall effect of this phenomenon is that the provided predictions are overconfident [118].

Focal Loss Many solutions have been proposed to mitigate the miscalibration issue of the CE, one of them consists in adjusting the weight of the contribution of a sample to the overall loss based on the network's predicted probability for its GT class [116], as done through the Focal Loss [91]:

$$\mathscr{L}_{\text{focal}} = -\sum_{m=1}^{M}\sum_{k=1}^{K} t_{m,k}(1-\hat{t}_{m,k})^{\gamma}\log(\hat{t}_{m,k}) \tag{6.6}$$

where γ is a hyperparameter controlling the rescaling strength.
Impact on the class separation: the hyperparameter γ enables tuning the magnitude of the weight rescaling, effectively bringing the loss value for correctly classified samples near zero and therefore mitigating the miscalibration and the growth of R^2.

Loss proposals for controlling the class separation

Besides the loss functions presented till here, it is clearly possible to design novel ones which, as is the case of the Focal Loss, provide hyperparameters allowing to control directly or indirectly the class separation. We make two proposals along this line and plot them in Fig. 6.4 to illustrate how they work.

MSE with a compressed sigmoid If we formalize our problem as a regression task we can use the Mean-Squared Error (MSE) loss function and compute it between ground truth values $l_m \in \{-1, 1\}$ and the output of the network passed through a sigmoid rescaled in the $[-1, 1]$ range. In order to have a hyperparameter that enables tuning the effect of our loss function we modify the slope of the sigmoid using a factor c obtaining the overall loss:

$$\mathscr{L}_{\text{MSE}} = \sum_{m=1}^{M}(\hat{s}_c(\sigma_m) - l_m)^2 \qquad \text{with} \quad \hat{s}_c(\sigma_m) = \frac{2}{1+e^{-c\sigma_m}} - 1 \tag{6.7}$$

(a) Compressed sigmoid

(b) Hinge loss trend

Figure 6.4: (a) Increasing c in the MSE compressed sigmoid transforms it into a Heaviside step function: the loss is zero when the output score has the correct sign. (b) H loss trend for positive ($l_m = 1$) and negative ($l_m = -1$) pairs ($\delta = 1$).

Impact on the class separation: the hyperparameter c tunes the loss value associated with different scores. With higher c values the sigmoid is more horizontally compressed becoming more and more similar to the Heaviside step function. The consequence is that the loss of already correctly classified samples goes towards zero.

Hinge Loss with controllable margin Instead of using a hyperparameter that allows us to *only decrease* the impact of the already correctly classified values on the overall loss function, as is the case of the previous proposal and of the Focal loss, we consider directly zeroing it when a specified condition is met. In particular, we assume that the score σ_m produced by our network can take positive or negative values indicating whether the network is more confident for the *same* or *different* class. We use zero as the threshold and impose a margin δ around it. Incorrectly classified samples, together with correctly classified ones that lie within the margin, linearly contribute to the overall loss, while the impact of the others is null, which means that the loss cancels out for $\sigma_m > \delta$ for positive samples and $\sigma_m < -\delta$ for negative ones. This way it is possible to directly control the distance between the two classes by choosing a value for δ. This formulation corresponds to a hinge loss applied on the scalar score σ_m:

$$\mathscr{L}_{\text{H}} = \sum_{m=1}^{M} \max(0, \delta - l_m \sigma_m) \qquad \text{with} \quad l_m \in \{-1, 1\} \tag{6.8}$$

6.2.5 The choice of a learning objective

Before embarking on a comprehensive quantitative analysis able to draw the state-of-the-art of fine-tuning-free OOD detection, we undertake a preliminary analysis to understand which of the learning objectives that can be used for ReSeND provides the best performance. In particular, for simplicity, we consider a subset of the settings of the intra-domain benchmark track that we will describe later, and for them, we analyze how the loss function shapes the feature space structure and we investigate how this

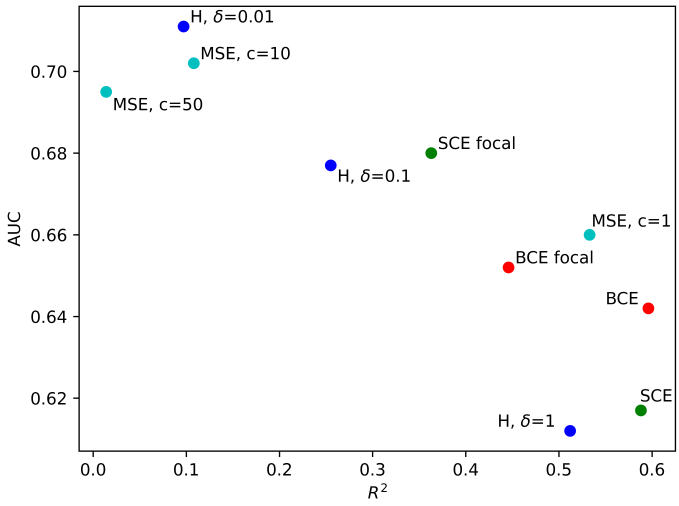
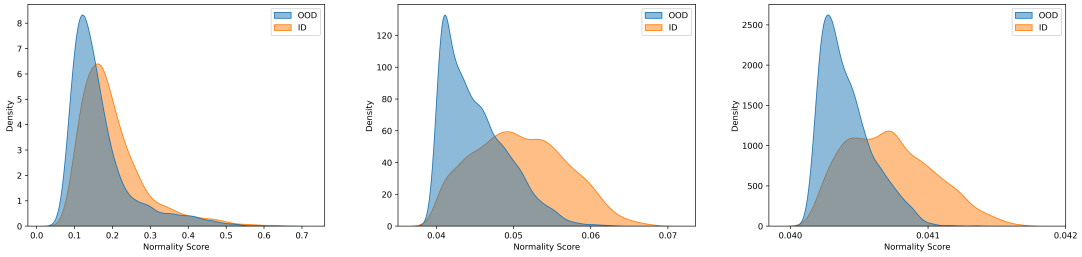


Figure 6.5: Analysis of the OOD performance of a relational reasoning-based model trained with different learning objectives: plot of the AUROC (AUC) against the R^2 index computed on the learned feature space. The scatter-plot shows that lower AUROC results are generally associated with higher R^2 values. This implies that a stronger class separation negatively impacts generalization



(a) SCE (b) H, δ=0.1 (c) H, δ=0.01

Figure 6.6: Example of distributions of normality scores for ID and OOD samples provided by models trained with different loss functions. The Hinge loss with low margin pushes the model to provide more conservative scores, which are very close to each other (see the horizontal axis' scale) but more discernible.

structure impacts the final OOD detection performance. We report average AUROC results plotted against the corresponding R^2 index values in the scatter plot of Fig. 6.5.

The plot highlights a **clear trend**: learning objectives imposing a higher inter-class separation generally provide lower performance. This behavior may seem counter-intuitive, as it means that loss functions that push for a lower and less clear separation of the features belonging to different classes obtain better overall performance. In order to understand why this happens we must keep into account the fact that we are measuring performance in a fine-tuning-free scenario. This means that the samples processed

by our models may belong to semantic classes that have never been seen by the model during training. From this point of view, in each of the OOD detection tasks that we consider, both *support* and *test* samples are equally novel to the model used to process them. As a result, the models that obtain better results are those that generalize better the knowledge learned during training. Therefore, we can reformulate the conclusion just drawn from the plot and affirm that what it shows is that the models that generalize better are those that impose a lower inter-class separation; this is not an entirely unexpected outcome [71].

The described phenomenon is even more evident when focusing on one of those loss functions that provide a hyperparameter that tunes the final inter-class separation. For example, if we consider the MSE, the impact of c is quite evident with better results obtained with higher values. Clearly, there is a limit to the performance gain that can be obtained in this way, because after a certain point, the features start losing their discriminative power (see the point referring to MSE with $c = 50$).

An alternative visualization of the impact of reducing the inter-class separation on the final OOD detection performance can be obtained by visualizing the distribution of normality score values for ID and OOD data. We do this exercise in Fig. 6.6, which clearly shows that, when using the Hinge loss with a small margin, the predictions of the network become more conservative but at the same time they simplify distinguishing between the two classes.

Given the results recorded in this preliminary analysis, in the rest of the chapter we will focus only on the two ReSeND variants that provide the best results, which are the one with MSE with $c = 10$, indicated simply as *ReSeND* from now on, and the one with the Hinge Loss with $\delta = 0.01$, indicated as *ReSeND-H*.

6.3 A benchmark and framework for OOD detection

6.3.1 Benchmark definition

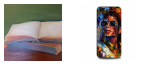
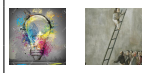
Starting from [50], which for the first time introduced the OOD detection task in deep learning-based CV research, most of the papers focusing on this problem have used evaluation benchmarks based on quite small datasets, both in terms of image resolution and number of classes: CIFAR [73] and MNIST [78]. Empirical results evaluated with this protocol, however, were found to not transfer to large-scale settings by Huang et al. [54], who thus proposed to discard that benchmark in favor of a larger scale one. This novel evaluation bed exploits ImageNet1k [34] as an In-Distribution dataset, with its train split used as the *support* set, and its validation split as the ID part of the *test* set. Subsets of iNaturalist [156], SUN [170], Places [187], and Textures [29] datasets are then used as sources of test OOD samples. While this benchmark certainly solves the scale problem of previous ones, we argue that the results it provides may still not perfectly reflect the real-world performance of the analyzed methods. Indeed, its definition involves ID and OOD data which may differ not only in terms of semantic class but also of data type (e.g. objects vs scenes or textures) and visual domain (these are *far OOD* settings). However, real-world problems (e.g. an autonomous vehicle confronting an uncommon object on its way) are much more difficult as they involve ID and OOD test samples representing objects of the same type, coming from the same visual domain, and differing only in terms of semantic class. The direct consequence of using the benchmark proposed in [54] is that the OOD detection tasks it defines are not only a bit unrealistic but also not particularly difficult, as highlighted by the exceptionally good results obtained by most methods on them [111].

Given these considerations, we propose a novel benchmark following the path proposed in [18, 3], where ID and OOD test samples are of the same type and come from the same visual domain. We design a large and comprehensive evaluation bed, composed of an intra-domain track and a cross-domain one.

Intra-domain analysis

This track is designed to evaluate specifically the semantic novelty detection ability of the analyzed models, which means that the *support* and *test* sets belong to the same visual distribution and the only shift considered is a semantic one. The track is built on top of 5 different datasets, chosen in order to represent a wide variety of different categorization tasks. Tab. 6.1 shows some examples of images coming from our benchmark and visually describes how the various OOD detection tasks are built. The available samples for each class of each dataset are divided into a train and a test split. Each OOD detection task is then built by randomly dividing the classes of the considered dataset into two groups: the train samples of ID classes are used as the *support* set, while the test samples for both ID and OOD classes compose the *test* set. In order to

Table 6.1: Examples of images and settings definition for our OOD detection benchmark. As a reference we also present the dataset benchmark from [54] where the OOD samples are drawn from a different dataset than the ID samples.

Setting	**Support (ID)**	**Test**	
		ID	**OOD**
Intra-domain	Textures 		
	PatternNet 		
	SUN 		
	Stanford Cars 		
	Real (DomainNet) 		
Cross-domain	Real (SS) 	Painting 	
	No Painting (MS) 		
Benchmark from [54]	ImageNet-1k 		iNaturalist  SUN  Places  Textures 

improve the statistical significance of our analysis we repeat the random split of each dataset's classes into ID and OOD groups three times and report average results in our

tables. We introduce below the datasets that we have selected for this analysis.

Textures [29] is a dataset containing 5640 images of textural patterns belonging to 47 different classes. We randomly select 23 of them as ID classes and keep the remaining as OOD ones. We define train and test splits following the first fold provided by the original paper's authors for their cross-validation strategy, and merging train and validation data. This dataset has already been used in the OOD detection literature, as part both of the standard benchmark initially proposed by Huang *et al.* [54] introduced before, and of a benchmark defined similarly to ours [18], but smaller.

PatternNet [192] is a dataset of aerial high resolution images that contains 38 classes with 800 images each. We select 19 classes as ID and keep the others as OOD. We use the train-test split provided by the original authors. To the best of our knowledge, this is the first time that this dataset is used as part of an OOD detection benchmark. We chose it as remote sensing categorization is an important task for many real-world applications and at the same time because it is based on images that differ significantly from the object-centric ones that are most used for lab research on visual categorization.

SUN [170] is a scene database containing 397 classes and 130k images in total. We select 198 classes as ID, and keep the rest as OOD. We use the train-test split provided by the original authors. This dataset has already been used in the OOD detection literature [54], but this is the first time it is used with a class split in ID and OOD groups.

Stanford Cars [72] is a dataset designed for fine-grained car classification. It includes more than 16k images divided into 196 classes. We adopt the train-test split provided by the original authors. We select 98 classes as ID and use the remaining 98 as OOD. This dataset has already been used as part of an OOD detection benchmark [111], but only as a whole to define ID classes.

DomainNet [127] is a large-scale dataset of common objects from 6 different visual domains. Considering that it contains object-centric images it is the dataset with the most similar object type to ImageNet1k. We use it in a similar way to the one proposed in [18], but extend its use to all of its 6 domains. Moreover, we go from 50 to 100 classes, chosen using the Natural Language Toolkit [7] to identify the classes having the smallest semantic overlap with ImageNet1k's. From those 100 classes we randomly select 50 as ID, and keep the others as OOD. We use the original authors' provided train-test splits. In the intra-domain experiments, both the *support* and the *test* data come from the same visual domain. There are therefore 6 different intra-domain tasks defined on DomainNet for which we directly report the average results in our tables.

Cross-domain analysis

In many open-world deployment scenarios, it is impossible to avoid the occurrence of visual domain shifts, which may happen simultaneously with semantic ones. We thus consider also this situation in our benchmark, as we want to perform an analysis as relevant as possible and that allows us to make considerations on what real-world performances should be expected when adopting a specific approach rather than another.

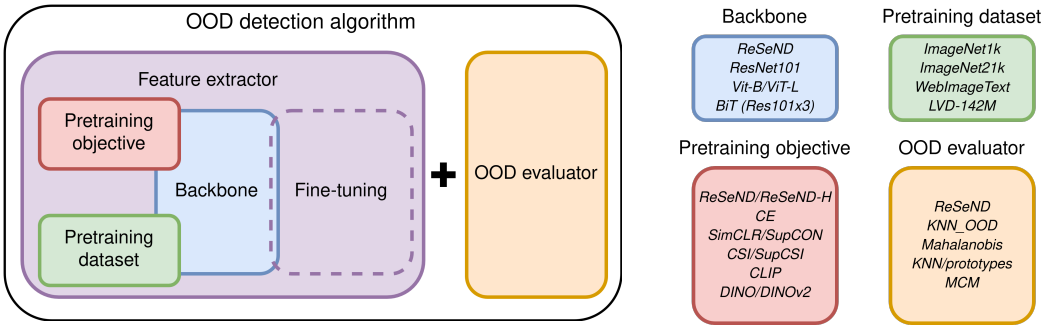


Figure 6.7: An OOD detection algorithm is composed of a feature extractor and an OOD evaluator. The former is obtained by choosing a backbone which is usually pre-trained on a dataset with a specific objective and can be further fine-tuned on the ID support data. The latter defines how to compute a normality score using the learned representations. The right part of the figure illustrates possible choices for each component.

Specifically, we consider cases in which there is a visual distribution shift between the *support* and the *test* set, which means that OOD test samples differ from ID *support* ones both in terms of appearance and semantics. This double difference makes their separation from ID test samples even more difficult as even the latter come from a different visual domain w.r.t. *support* data. We consider two settings inside this cross-domain benchmark track, both built on top of the DomainNet [127] dataset, for which we use the same ID-OOD and train-test splits of the intra-domain case.

Single-Source → Single-Target (SS → ST). We adopt a single visual domain's ID train data as the *support* set while using another domain's *test* set. Considering that DomainNet contains 6 domains, we can create 30 different settings and we report average results in our tables.

Multi-Source → Single-Target (MS → ST). We use 5 domains' ID train data together to build the *support* set and use *test* data from the kept-out domain. There are 6 different settings that can be defined in this way, once again we report average results over the available settings in our tables.

6.3.2 Framing OOD algorithms

The literature focusing on the OOD detection task is quite broad and includes papers proposing a wide variety of algorithms that can differ significantly, as we have seen when presenting the various families of methods in Chap. 2. Given that we are interested in drawing a comprehensive picture of the state-of-the-art, we include in our experimental analysis a large number of methods and we propose here to collect all of them under a common framework, which simplifies their comparison.

We consider an OOD detection algorithm to be composed of two main parts: a *feature extractor* and an *OOD evaluator*. The former is tasked to provide representations for the samples in analysis, and the latter to use those representations to estimate the

normality of test samples. We illustrate this structure in Fig. 6.7 left, while on the right we list some of the choices that can be taken for each component, in particular those that allow us to build fine-tuning-free OOD algorithms. We proceed by providing more details on each of these two components and listing and contextualizing the methods that we include in our experimental analysis.

OOD evaluator

The *evaluator* is the component on which OOD detection algorithms generally base their individuality and on which they differ the most. It defines how the *normality* of a test sample is evaluated and this definition is the core of the task. Indeed, OOD detection in general requires the ability to extract a concept of *normality* from the *support* data and to encode it in a way that supports performing comparisons, so that it is possible to estimate deviations from it. The traditional way to perform these steps involves training a model on the support data so that the *normality* is automatically encoded in the model weights, and later defining an *evaluator* by looking for differences in the treatment of normal and abnormal data by the model itself. Any fine-tuning-free approach, on the other hand, requires the *normality encoding* to be explicit: *support* samples are used to create a *normality compendium* which is then used to identify OOD samples by using an *evaluator* exploiting some kind of distance metric, or other concept-matching approaches.

Fine-tuning-free evaluators The easiest way to compare test samples with support ones is to compute distances between their representations. A really naïf approach to OOD detection is thus to use the average distance from the K nearest support neighbors (**KNN**) as a measure of normality. A simple variant to this strategy has actually been proposed in **KNN_norm** [149], by introducing as the only addition a feature normalization step applied before computing Euclidean distances. Clearly, these strategies have the disadvantage of requiring storing the feature representations of all the *support* samples as the *normality compendium*. This can represent a serious limitation in terms of memory and computational footprint when the *support* dataset has high cardinality. A possible mitigation of this issue consists in reducing the cardinality of the *normality compendium*, by computing per-class feature averages in order to obtain *known* class **prototypes**. Indeed, using the distance from the nearest prototype to compute a normality score, allows one not only to reduce the computational cost, but potentially also to avoid making wrong decisions that may be influenced by the presence of outliers in the *support* data. Besides the Euclidean distance, other metrics could be used with the same comparison purpose. One option consists in using the **Mahalanobis** [79] distance, which relies on a modeling of the *normality* through multivariate Gaussian distributions fitted on support classes.

Other approaches rely on specific architectures or learning objectives adopted in the pre-training phase. We have already seen that one possibility is to learn a similarity

metric directly in the pre-training phase through a relational reasoning objective, as done by **ReSeND** and **ReSeND-H**. Another option consists in adopting a multi-modal pre-training like CLIP [129] and exploiting an auxiliary modality to create the *normality compendium*. This is the path followed by Maximum Concept Matching (**MCM**), which uses known class prototypes obtained by using the *names* of those classes as part of textual prompts provided as input to the CLIP's text encoder. These text-originated prototypes are then compared directly with visual representations of test samples through cosine similarity. This approach has both advantages and disadvantages, on one side it allows to avoid completely the need for visual *support* samples, on the other side it needs to access names for known classes, which should also belong to object types similar to the ones met during the pre-training.
Note: both **KNN_norm** and **Mahalanobis** have been proposed as fine-tuning-based OOD detection methods, respectively in [149] and [79]. However, the distance-based evaluators they propose do not really have requirements linked to their use only with fine-tuned models. We thus follow Lu *et al.* [101] in using them also as fine-tuning-free approaches.

Fine-tuning-based evaluators Our main focus in this chapter is to draw a comprehensive picture of the state-of-the-art of the *fine-tuning-free* OOD detection field. This kind of picture, however, cannot be considered complete if it does not involve some comparisons with the orthogonal fine-tuning-based research line. We thus include also some fine-tuning-based approaches in our analysis, by selecting some examples that we find relevant from a literature that, in this field, is particularly extensive.

First of all, we adopt the standard baseline that is used in this context, the Maximum Softmax Probability (**MSP**). It is based on the idea that when a model is trained for classification on a bunch of semantic classes, after deployment it will provide predictions with higher confidence for ID samples than for OOD ones. Even if this is not always the case because of the overconfidence of classifiers [118], this approach is still a good baseline that should never be neglected. Various approaches have then been proposed to improve over MSP while adopting the same simple fine-tuning strategy. **ReAct** [148] uses a rectification operation on neural activations which is designed to avoid spikes in their values that could cause high-confidence predictions for OOD samples. On a similar line, **ASH** [35] filters out a majority of the activations keeping only the amount corresponding to a specific p-percentile. Both ReAct and ASH support a number of evaluators, ranging from the standard MSP to more advanced approaches, like the one based on energy scores [95].

The algorithms listed till here belong to the *post-hoc* family, as they can be applied to any model that has been trained for classification with a standard CE-based objective. As was the case for fine-tuning-free approaches, however, even in the fine-tuning-based world some methods require specific training paradigms. Among them, a promising line of research proposes to adopt hybrid generative-discriminative models that are able to perform classification and likelihood estimation jointly. From this family, we

adopt a **Flow**-based approach inspired by the OSR method OpenHybrid [180].

Feature extractor

The feature extractor may seem a component of only secondary importance in the definition of an OOD detection algorithm. However, the **quality of the representations** used by an OOD evaluator significantly impacts the overall performance of the algorithm, making the choice of how to choose or build a good feature extractor particularly important. In most cases, the starting point is a standard neural network backbone trained with a representation learning objective on a specific dataset. The choice of the **network architecture**, but even more of the **pre-training objective** and **pre-training dataset**, is sometimes of minor significance for fine-tuning-based approaches, that in one way or in another will overwrite most of the knowledge encoded by the pre-trained model, but it is of primary importance for fine-tuning-free strategies.

Pre-training dataset In our analysis, we consider two macro-groups of OOD detection algorithms, differing on the basis of the **pre-training dataset**. On one side we have approaches that adopt a pre-training based on **ImageNet1k**, which with its 1.2M of samples spanning 1000 semantic classes, has been the standard pre-training dataset for a long time. Pre-trainings of this kind were primarily exploited for transfer-learning and thus always followed by a fine-tuning phase. Also ReSeND is trained on this dataset, exploiting the very large number of possible *same* and *different* class sample pairs that its size enables creating. As ReSeND does not require fine-tuning, we evaluate also other ImageNet1k's pre-trainings to understand if they support OOD detection without fine-tuning. The second macro-group is intended to accommodate more modern representation learning paradigms designed for extremely large-scale data collections. Indeed, the exponential growth in size of the deep neural networks in the last years has made even ImageNet1k too small to perform an effective training. This necessitated on the one hand the construction of ever larger datasets and on the other hand the definition of sophisticated large-scale training strategies. We include in this group algorithms using as pre-training dataset **ImageNet21k** with its 14M of images, **WebImageText**, the dataset behind CLIP [129], which counts 400M of image-text pairs, and **LVD-142M**, the unsupervised but curated dataset used for the training of DINOv2 [122].

Pre-training objective It is the second element necessary to build a good feature extractor, and from this point of view, it is impossible to not adopt the standard **CE** as a reference. We also investigate contrastive-based self-supervised and supervised objectives like **SimCLR** [26] and **SupCon** [66]. We consider then their variants designed specifically for semantic novelty detection thanks to the introduction of semantically shifting transformations *i.e.* **CSI** and **SupCSI** [152]. We also take into account some more recent self-supervised approaches, in particular **DINO** [24], which uses contrastive learning

and knowledge self-distillation, and its variant **DINOv2** [122] that adds to the other objectives a masked-image modeling one. The latter, by adopting an extremely large-scale training procedure, both in terms of data cardinality and computation distribution, is able to provide general-purpose representations of quality high enough to consider the model as the first vision-only foundation model. As last pre-training objective we consider **CLIP** [129], which uses a vision-language contrastive learning objective designed to train jointly a text and a vision encoder so that they provide similar representations for images and textual descriptions in pairs. This objective, when applied to a sufficiently large dataset such as WebImageText, is able to provide a vision-language multi-modal foundation model.

Network architecture As far as this element is concerned we try to cover a set of options that allow us to provide a picture as comprehensive as possible. We start by considering both CNN and transformer-based backbones since these are the two most common architectures adopted in the literature. Specifically, from the first group we use a **ResNet101** [48] with 44M of parameters and the much larger **BiT** [70] with 380M. The latter is a wide ResNet with some changes introduced to make its representations more transferable, for example by substituting the Bath Normalization layers with Group Normalization [169] and Weight Standardization [128] ones. From the second family, we consider both a **ViT-B** with 86M of parameters and a **ViT-L** [37] with 307M. The only exception to this list is done for **ReSeND**, which is based on its own architecture designed to support relational reasoning.

Algorithms included in our experimental analysis

In general, we do not consider all possible combinations for pre-training datasets, objectives, and network architectures, but only a limited set of relevant ones. In particular, the smaller networks are trained with ImageNet1k, while for the big ones larger datasets are preferred. When possible and especially for large models we rely on publicly available checkpoints such as the one provided by PyTorch and HuggingFace. A similar selection strategy based on relevance and compatibility is performed to pair feature extractors and OOD evaluators.

Table 6.2: Training-free OOD detection results. We consider the experiments on ImageNet1k and those on larger datasets as separate settings, displayed in the two horizontal subparts of the table (top and bottom). For each of them, we use the bold font to highlight the best result per column

Pretraining Dataset	Backbone	Pretraining Objective	OOD Evaluator	Intra-domain track												Cross-domain track			
				Textures		PatterNet		SUN		Stanford Cars		DomainNet Intra		AVG		SS → ST		MS → ST	
				AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓
ImageNet1k	ReSeND	ReSeND	ReSeND	0.667	0.881	0.871	0.550	0.582	0.911	0.494	0.947	0.612	0.900	0.645	0.838	0.560	0.930	0.581	0.924
		ReSeND-H		0.679	0.867	0.909	0.463	0.586	0.910	0.496	0.951	0.628	0.894	0.660	0.817	0.553	0.933	0.575	0.927
	ResNet101	CE	KNN_norm	0.758	0.789	0.936	0.296	0.639	0.874	**0.571**	**0.929**	0.696	0.815	0.720	0.740	**0.576**	0.915	**0.617**	0.888
			Mahalanobis	0.641	0.885	0.850	0.550	0.574	0.908	0.544	0.936	0.609	0.893	0.643	0.835	0.540	0.934	0.555	0.929
			KNN	0.698	0.851	0.878	0.542	0.601	0.899	0.566	0.932	0.650	0.871	0.678	0.819	0.551	0.936	0.580	0.924
			prototypes	0.667	0.884	0.827	0.684	0.615	0.887	0.540	0.938	0.609	0.906	0.652	0.860	0.537	0.942	0.540	0.941
		SimCLR	KNN	0.504	0.941	0.663	0.877	0.501	0.953	0.500	0.945	0.498	0.950	0.533	0.933	0.488	0.954	0.487	0.955
			prototypes	0.496	0.948	0.564	0.956	0.498	0.951	0.498	0.941	0.499	0.947	0.511	0.949	0.500	0.950	0.506	0.951
		SupCon	KNN	0.563	0.931	0.838	0.653	0.510	0.952	0.513	0.945	0.536	0.939	0.592	0.884	0.503	0.947	0.509	0.946
			prototypes	0.528	0.948	0.647	0.872	0.492	0.953	0.503	0.947	0.515	0.947	0.537	0.933	0.502	0.949	0.506	0.950
		CSI	KNN	0.652	0.870	0.884	0.553	0.575	0.920	0.520	0.943	0.583	0.914	0.643	0.840	0.516	0.944	0.532	0.936
			prototypes	0.613	0.903	0.790	0.782	0.546	0.937	0.499	0.948	0.543	0.937	0.598	0.902	0.511	0.949	0.510	0.948
		SupCSI	KNN	0.618	0.896	0.877	0.528	0.578	0.917	0.533	0.939	0.604	0.911	0.642	0.838	0.525	0.942	0.543	0.934
			prototypes	0.590	0.912	0.783	0.708	0.561	0.928	0.511	0.944	0.565	0.930	0.602	0.885	0.518	0.944	0.517	0.945
	ViT-B	CE	KNN_norm	0.725	0.804	0.914	0.380	0.628	0.892	0.575	0.928	0.686	0.836	0.705	0.768	0.562	0.922	0.597	0.905
			Mahalanobis	0.610	0.915	0.795	0.742	0.590	0.931	0.542	0.940	0.602	0.905	0.628	0.887	0.530	0.941	0.542	0.934
			KNN	0.595	0.919	0.842	0.644	0.552	0.932	0.550	0.941	0.620	0.901	0.632	0.867	0.522	0.943	0.546	0.932
			prototypes	0.538	0.916	0.810	0.665	0.568	0.911	0.530	0.945	0.592	0.898	0.608	0.867	0.518	0.943	0.520	0.938
		DINO	KNN	**0.763**	**0.758**	**0.956**	**0.227**	**0.664**	**0.838**	0.566	0.929	**0.700**	**0.811**	**0.730**	**0.713**	0.575	**0.906**	0.616	**0.869**
			prototypes	0.764	0.775	0.932	0.325	0.683	0.801	0.530	0.938	0.664	0.843	0.714	0.736	0.558	0.918	0.573	0.907
ImageNet21k	ViT-L	CE	KNN_norm	0.726	0.810	0.812	0.674	0.726	0.828	0.571	0.933	0.699	0.793	0.707	0.808	0.600	0.902	0.622	0.881
			Mahalanobis	0.706	0.830	0.664	0.843	0.783	0.746	0.553	0.939	0.668	0.802	0.675	0.832	0.590	0.903	0.626	0.868
			KNN	0.710	0.850	0.809	0.675	0.711	0.859	0.565	0.934	0.689	0.832	0.697	0.830	0.586	0.922	0.611	0.903
			prototypes	0.675	0.865	0.758	0.758	0.782	0.791	0.553	0.930	0.676	0.855	0.689	0.840	0.563	0.931	0.588	0.915
	BiT (Res101x3)	CE	KNN_norm	**0.832**	**0.635**	0.938	0.303	0.760	0.775	0.582	0.923	0.724	0.766	0.767	0.680	0.589	0.899	0.643	0.845
			Mahalanobis	0.709	0.871	0.756	0.723	0.653	0.891	0.524	0.942	0.614	0.892	0.651	0.864	0.545	0.935	0.576	0.917
			KNN	0.804	0.729	**0.950**	**0.254**	0.734	0.808	0.578	0.922	0.721	0.788	0.757	0.700	0.581	0.912	0.636	0.864
			prototypes	0.740	0.812	0.897	0.487	0.748	0.787	0.545	0.932	0.668	0.853	0.720	0.774	0.550	0.928	0.567	0.919
WebImageText	ViT-L	CLIP	MCM	0.701	0.861	0.776	0.851	0.766	0.742	0.517	0.944	**0.817**	0.698	0.716	0.819	**0.817**	**0.698**	**0.817**	**0.698**
LVD-142M		DINOv2	KNN	0.795	0.735	0.911	0.378	0.775	0.736	**0.720**	**0.848**	0.791	**0.684**	**0.798**	0.676	0.676	0.836	0.713	0.808
			prototypes	0.803	0.722	0.904	0.430	**0.834**	**0.593**	0.626	0.902	0.784	0.687	0.790	**0.667**	0.670	0.851	0.722	0.786

6.4 Benchmarking fine-tuning-free OOD detectors

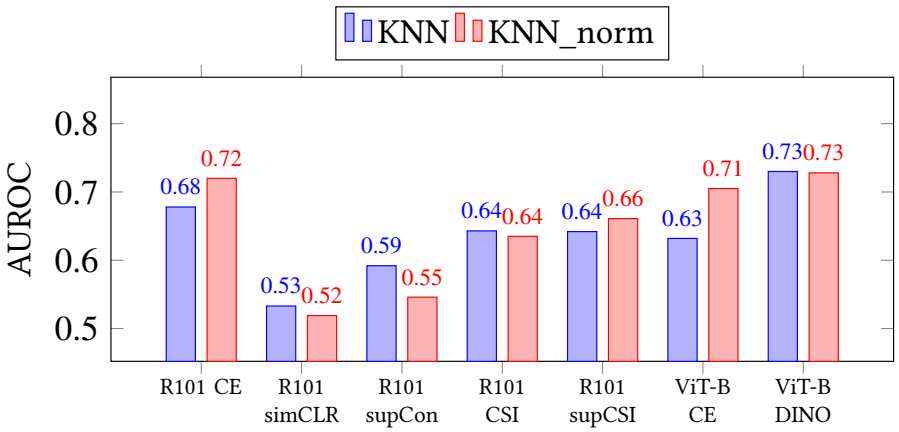


Figure 6.8: KNN vs KNN_norm performance for ImageNet1k-based pre-trainings on the intra-domain track

In this section, we report and describe the results that the OOD detection algorithms

Table 6.3: Analysis of the CLIP visual encoder when using KNN and prototypes evaluators vs the CLIP text encoder exploited by the MCM evaluator. We also report the same DINOv2 results presented in Tab. 6.2 as reference.

Pretraining Dataset	Backbone	Pretraining Objective	OOD Evaluator	Intra-domain AVG AUROC↑	Intra-domain AVG FPR95↓	Cross-domain SS → ST AUROC↑	Cross-domain SS → ST FPR95↓	Cross-domain MS → ST AUROC↑	Cross-domain MS → ST FPR95↓
WebImageText	ViT-L	CLIP	MCM	0.716	0.819	**0.817**	**0.698**	**0.817**	**0.698**
			KNN	**0.812**	**0.658**	0.648	0.915	0.695	0.894
			prototypes	0.798	0.699	0.636	0.928	0.677	0.921
LVD-142M		DINOv2	KNN	0.798	0.676	0.676	0.836	0.713	0.808
			prototypes	0.790	0.667	0.670	0.851	0.722	0.786

that we have described obtain when they are applied to our novel large-scale benchmark. We evaluate those algorithms according to the OOD detection metrics presented in Sec. A.3.2.

6.4.1 Main results

We report the main results in Tab. 6.2. We discuss them in detail below focusing first on methods of the top group, based on pre-trainings of smaller size, and then on methods of the second one, where also foundation models are considered.

ImageNet1k-based pre-trainings

The **two top-performing algorithms** are clearly KNN_norm applied on a CE-based ResNet101, and KNN applied on a DINO-based ViT-B. Both these results are particularly significant for their own reasons. The first is an interesting finding because **the original paper proposing KNN_norm [149] did not consider applying it without fine-tuning** the model on the *support* data, thus the exceptional performance highlighted here may be novel even for the original authors. Moreover, the significant difference in performance between KNN_norm and standard KNN when applied on the same models, indicates that **the normalization trick of KNN_norm contributes significantly** to the final performance of the approach. The reason is that when using CE to train the model, the loss is non-zero even in the case of correct predictions, which pushes the feature norms to grow indefinitely during training. The consequence of this behavior is that the norms lose their relevance, thus discarding them through the normalization step means discarding a misleading component. The same behavior is not recorded when using other learning objectives which do not influence the norms of the features in the same way. These claims are confirmed by Fig. 6.8 which contains a direct comparison between KNN and KNN_norm with different pre-trainings. For what concerns **DINO**, the interesting thing to note is that its good results are obtained thanks to **representations learned in a completely self-supervised fashion**. The clear improvement between DINO and older self-supervised methods like SimCLR testifies to the significant progress made by the research in this field in the last years and paves

the way for even stronger improvements that can be obtained by scaling self-supervised strategies.

The performances of **ReSeND** in its two variants are quite interesting, even if they do not reach the state-of-the-art among methods trained only on ImageNet1k and with a comparable number of parameters. Indeed, **ReSeND-H** is the top-performing method among those that perform comparisons using known class prototypes as *normality compendium*, a strategy that is far less computationally expensive than those based on KNN. In particular, ReSeND outperforms all contrastive-based approaches, which, despite some differences, adopt a similar comparison-based logic in the training phase.

By comparing intra-domain and cross-domain results, we immediately notice that average performances are significantly lower in the second case, highlighting that all the considered pre-training strategies provide representations that are clearly domain-dependent.

Larger pre-trainings

The first two rows of this group show that applying a *traditional* CE-based pre-training, even if on top of a larger supervised dataset like ImageNet21k, does not seem to improve much the performance w.r.t. the ImageNet1k case. A greater improvement can be obtained with a pre-training designed explicitly for transfer learning as is the case of **BiT**. Still, even in this case, the **cross-domain results are pretty poor**, highlighting that the ImageNet-based representation learning approaches are still significantly affected by the visual distribution of the training data. A completely different picture is drawn by the results of **MCM**. This approach, thanks to its multimodal vision-language pre-training, obtains exceptionally good results when the data type of test samples is the same as that of the pre-training ones, i.e. object-centric images, such as the ones in DomainNet. Results are not as good, instead, for very different data types (e.g. for textures in DTD), image points of view (PatternNet), and fine-grained classification cases (Stanford Cars), showing that the CLIP image encoder, or the corresponding text encoder, may not be able to extract significant representations for the considered class types and names. Overall, however, even if its results in the intra-domain benchmarks are not the top-performing ones, this method is particularly relevant as its cross-domain performance represents the state-of-the-art by a large margin. Nevertheless, it should be noted that there's an **important difference** between how the CLIP-based MCM strategy and all the other ones use the *support* data to represent the *normality*. Indeed, MCM extracts the normality for any given OOD detection problem from textual inputs built on class names, while all the other approaches use visual *support* data. As a consequence, MCM cannot be influenced by any visual distribution shift appearing between *support* data and *test* data. In order to disentangle the impact of the text encoder from that of the visual encoder, we perform an additional analysis in Tab. 6.3, in which we test also the latter alone using the KNN and prototypes evaluators. What we can notice is that in the intra-domain case discarding the language cues brings a significant

improvement in performance. On the other hand in the cross-domain case the effect is the opposite, and the CLIP's vision encoder is outperformed by DINOv2. These results highlight the fact that language can represent a significant asset in many scenarios, but blindly relying on it in every setting may be deleterious.

In the intra-domain benchmark, the top overall performance is obtained by DINOv2, highlighting again the robustness of self-supervised representation learning approaches, especially when applied at scale. Interestingly, this pre-training obtains extremely similar results when tested with KNN or ID prototypes, showing that it is able to build compact known class clusters for which the centroids are not far away from the cluster boundaries. This capability may represent a significant selling point, as it allows to discard the KNN approach in favor of the prototypes-based one, decreasing significantly the computational cost, without sacrificing good performance. If we consider unimodal methods only, even in the cross-domain case this method obtains the top-performing results, highlighting the greater visual domain-invariance of DINOv2 w.r.t. its competitors.

To sum up, **DINOv2** obtains the **best overall performance** between vision-only approaches and this is the **first time** that the fine-tuning-free OOD detection abilities of this model are proven. This outcome, however, is not completely unexpected, as it fits perfectly in the context of the wide applicability of the representations provided by foundation models, further proving their potential to induce a significant change in the research field.

6.4.2 Comparison with Fine-tuning-based state-of-the-art

As previously discussed, a comprehensive picture of the state-of-the-art in the fine-tuning-free OOD detection field cannot be considered complete if it does not include a comparison with fine-tuning-based methods that is necessary to understand in which situations it is still preferable to adopt one solution instead of the other.

We thus consider models using ViT-B, ViT-L, and BiT as the backbone, originally pre-trained on ImageNet1k, ImageNet21k, and LVD-142M, and fine-tune them with the standard CE learning objective or with the Flow-based hybrid objective of [180]. Once fine-tuned, we test these models with the fine-tuning-based evaluators listed in Sec. 6.3.2.

Results on our OOD detection benchmark

We test the fine-tuning-based OOD detection methods and compare their results with the top-performing fine-tuning-free approaches in Tab. 6.4.

The results as a whole show how fine-tuning provides a clear advantage in the intra-domain track (left part of the table), while in the cross-domain track (right part of the table) it produces mixed results. For instance, the AUROC of the fine-tuned DINOv2 KNN_norm is 0.748 for MS→ST, with a clear improvement over the corresponding

Table 6.4: Fine-tuning-free vs Fine-tuning-based OOD detection results on our benchmark. For each setting we use the bold font to highlight the best result per column.

Pretraining Dataset	Backbone	Pretraining Objective	OOD Evaluator	Intra-domain track												Cross-domain track			
				Textures		PatterNet		SUN		Stanford Cars		DomainNet Intra		AVG		SS → ST		MS → ST	
				AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓
training-free methods																**training-free methods**			
ImageNet21k	BiT (Res101x3)	CE	KNN_norm	**0.832**	**0.635**	0.938	0.303	0.760	0.775	0.582	0.923	0.724	0.766	0.767	0.680	0.589	0.899	0.643	0.845
			Mahalanobis	0.709	0.871	0.756	0.723	0.653	0.891	0.524	0.942	0.614	0.892	0.651	0.864	0.545	0.935	0.576	0.917
			KNN	0.804	0.729	**0.950**	**0.254**	0.734	0.808	0.578	0.922	0.721	0.788	0.757	0.700	0.581	0.912	0.636	0.864
			prototypes	0.740	0.812	0.897	0.487	0.748	0.787	0.545	0.932	0.668	0.853	0.720	0.774	0.550	0.928	0.567	0.919
WebImageText	ViT-L	CLIP	MCM	0.701	0.861	0.776	0.851	0.766	0.742	0.517	0.944	**0.817**	0.698	0.716	0.819	**0.817**	**0.698**	**0.817**	**0.698**
LVD-142M		DINOv2	KNN	0.795	0.735	0.911	0.378	0.775	0.736	**0.720**	**0.848**	0.791	**0.684**	**0.798**	0.676	0.676	0.836	0.713	0.808
			prototypes	0.803	0.722	0.904	0.430	**0.834**	**0.593**	0.626	0.902	0.784	0.687	0.790	**0.667**	0.670	0.851	0.722	0.786
training-based methods																**training-based methods**			
ImageNet1k	ViT-B	CE	MSP	0.793	0.748	0.985	0.063	0.771	0.789	0.867	0.604	0.816	0.732	0.847	0.587	0.620	0.910	0.708	0.861
			ReAct+energy	0.814	0.691	0.986	0.061	0.801	0.734	0.863	0.644	0.837	0.662	0.860	0.558	0.632	0.898	0.730	0.840
			KNN_norm	0.808	0.692	**0.992**	**0.043**	0.785	0.768	0.861	0.621	0.842	0.661	0.858	0.557	0.644	0.899	0.735	0.840
			ASH+energy	0.488	0.959	0.466	0.958	0.511	0.949	0.496	0.947	0.482	0.949	0.489	0.952	0.480	0.957	0.464	0.961
			Flow	0.785	0.753	0.992	0.048	0.776	0.764	0.829	0.736	0.833	0.679	0.843	0.596	0.636	0.900	0.730	0.847
		DINO	MSP	0.778	0.788	0.973	0.097	0.762	0.803	0.862	0.647	0.807	0.754	0.836	0.618	0.609	0.915	0.695	0.875
			ReAct+energy	0.799	0.741	0.977	0.090	0.785	0.772	0.853	0.728	0.830	0.697	0.849	0.606	0.627	0.904	0.717	0.862
			KNN_norm	0.786	0.725	0.986	0.073	0.759	0.801	0.824	0.746	0.825	0.695	0.836	0.608	0.638	0.897	0.712	0.863
			ASH+energy	0.519	0.940	0.493	0.939	0.497	0.951	0.500	0.944	0.508	0.948	0.503	0.944	0.503	0.950	0.509	0.948
			Flow	0.764	0.773	0.974	0.127	0.710	0.845	0.729	0.858	0.799	0.763	0.795	0.673	0.617	0.909	0.694	0.888
ImageNet21k	BiT (Res101x3)	CE	MSP	0.760	0.805	0.945	0.188	0.760	0.823	0.861	0.662	0.802	0.773	0.826	0.650	0.609	0.916	0.692	0.882
			ReAct+energy	0.787	0.750	0.941	0.181	0.783	0.784	0.859	0.652	0.828	0.681	0.840	0.609	0.622	0.909	0.716	0.865
			KNN_norm	0.797	0.730	0.986	0.077	0.789	0.765	0.857	0.676	0.842	0.653	0.854	0.580	0.640	0.900	0.721	0.843
			ASH+energy	0.780	0.751	0.869	0.615	0.778	0.768	0.852	0.652	0.813	0.695	0.818	0.696	0.612	0.911	0.711	0.860
			Flow	0.757	0.763	0.977	0.111	0.753	0.812	0.803	0.849	0.826	0.703	0.823	0.648	0.643	0.897	0.721	0.845
LVD-142M	ViT-L	DINOv2	MSP	0.820	0.732	0.969	0.092	0.797	0.767	0.910	0.408	0.844	0.686	0.868	0.537	0.665	0.878	0.723	0.851
			ReAct+energy	**0.842**	0.627	0.973	0.093	**0.826**	**0.681**	**0.919**	**0.322**	**0.867**	0.582	0.885	**0.461**	0.686	0.855	0.747	**0.831**
			KNN_norm	0.840	**0.616**	0.986	0.060	0.822	0.703	0.913	0.350	**0.867**	**0.574**	**0.886**	**0.461**	0.696	**0.847**	**0.748**	0.832
			ASH+energy	0.508	0.944	0.491	0.913	0.481	0.959	0.504	0.947	0.522	0.934	0.501	0.939	0.511	0.942	0.514	0.939
			Flow	0.813	0.683	0.978	0.096	0.808	0.726	0.884	0.534	0.862	0.600	0.869	0.528	**0.691**	**0.847**	0.735	0.858

Table 6.5: Fine-tuning-free vs Fine-tuning-based OOD detection results on the benchmark from [54]. ID dataset: ImageNet1k. The column titles indicate the OOD dataset. We use the bold font to highlight the best result per column and training setting.

Pretraining Dataset	Backbone	Pretraining Objective	OOD Evaluator	iNaturalist		SUN		Places		Texture		AVG	
				AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC	FPR95↓
training-free methods													
ImageNet21k	BiT (Res101x3)	CE	KNN_norm	0.935	0.376	0.841	0.696	0.826	0.717	0.911	0.397	0.878	0.547
			Mahalanobis										
			KNN	0.958	0.226	0.743	0.905	0.743	0.890	0.955	0.180	0.850	0.550
			prototypes	0.973	0.154	0.792	0.804	0.766	0.808	**0.963**	**0.169**	0.873	0.484
WebImageText	ViT-L	CLIP	MCM	0.950	0.283	**0.941**	**0.286**	**0.923**	**0.343**	0.831	0.625	0.911	0.384
			KNN	0.810	0.952	0.718	0.954	0.733	0.922	0.807	0.844	0.767	0.918
			prototypes	0.846	0.866	0.768	0.921	0.761	0.896	0.820	0.797	0.799	0.870
LVD-142M		DINOv2	KNN	0.922	0.333	0.808	0.742	0.828	0.696	0.859	0.561	0.854	0.583
			prototypes	**0.983**	**0.044**	0.916	0.372	0.894	0.449	0.906	0.400	**0.925**	**0.316**
training-based methods													
ImageNet21k	BiT (Res101x3)	CE	MSP	0.884	0.583	0.787	0.775	0.785	0.776	0.785	0.750	0.810	0.721
			ReAct+energy	0.909	0.717	0.839	0.775	0.830	0.739	0.771	0.916	0.837	0.787
			KNN_norm	0.932	0.424	0.825	0.746	0.814	0.758	0.940	0.228	0.878	0.539
			ASH+energy	0.936	0.453	0.852	0.673	0.834	0.677	0.843	0.748	0.866	0.637
			Flow	0.915	0.490	0.848	0.643	0.803	0.743	**0.977**	**0.114**	0.886	0.497
LVD-142M	ViT-L	DINOv2	MSP	0.931	0.408	0.831	0.687	0.825	0.694	0.833	0.657	0.855	0.611
			ReAct+energy	**0.970**	**0.157**	0.882	0.505	0.867	0.538	0.885	0.464	0.901	**0.416**
			KNN_norm	0.950	0.310	0.864	0.623	0.864	0.601	0.839	0.633	0.879	0.542
			ASH+energy	0.577	0.958	0.695	0.824	0.593	0.914	0.754	0.729	0.655	0.856
			Flow	0.960	0.240	**0.909**	**0.427**	**0.885**	**0.503**	0.864	0.602	**0.904**	0.443

not-fine-tuned result of KNN which is 0.713. However, the trend inverts if we consider

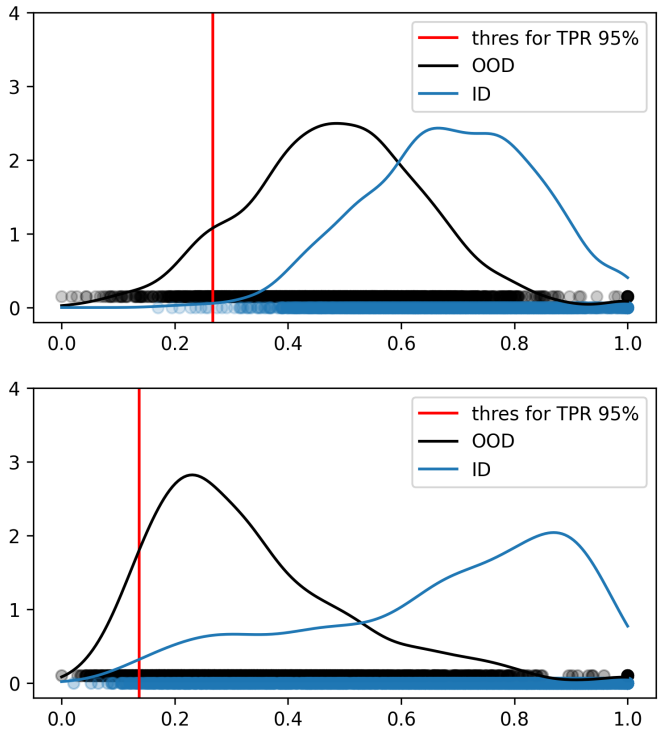


Figure 6.9: Distribution of normality scores for ID and OOD test samples on the All (MS) → Clipart (ST) set of the cross-domain track, using the pre-trained DINOv2 model (top) and the corresponding fine-tuned one (bottom). Note how the range of score values, represented by the blue-ID / black-OOD horizontal bars over the x-axis, become similar in the bottom figure. Despite the peaks of the two distributions moving apart, the separating threshold for TPR95 decreases, causing a worse FPR95 score.

FPR95 (for which lower is better), which is 0.832 for the fine-tuned KNN_norm and 0.808 for the not-fine-tuned KNN. A similar behavior holds for the SS→ST case. The interesting element here is that this *inconsistency* between AUROC and FPR95 results is a direct consequence of the use of a biased feature extractor resulting from the adoption of a fine-tuned model. Indeed, in this case, OOD samples are represented with features that have been adapted to ID *support* data. As a result, in the fine-tuned feature space the unknown classes appear close to the known ones, and specifically, they appear closer than what they used to do in the non-fine-tuned one. The direct consequence of this phenomenon is that the ranges of normality scores provided for ID and OOD test samples become more similar, as shown in Fig. 6.9 (see the black/blue colored bars over the horizontal axis), and thus even if the peaks of the two distributions move apart improving the AUROC score, a lower threshold has to be chosen to have a TPR of 95%, which leads to a worse FPR95 score.

Regardless of these details, in the cross-domain track, the best performance remains

that of the not-fine-tuned MCM that exploits the text-encoder. Overall, it is worth noting that when fine-tuning is performed, the effect of the size of the original pre-training datasets becomes less evident than in the fine-tuning-free case. Specifically, the difference between the best AUROC when using ImageNet1k or ImageNet21k/LVD142M can get up to 0.1 in the training-free case and it is instead below 0.06 in the fine-tuned case. As a final remark, we highlight that the very recent ASH approach performs well only when applied on top of convolutional backbones (BiT), while it obtains poor results when using transformer-based backbones, as was already noted in [183].

Results on MOS [54] benchmark

As mentioned in Sec 6.3, the most commonly used testbed in the recent OOD detection literature is the one proposed in [54], which covers only *far OOD* settings using ImageNet1k as ID data. In order to provide a more complete performance overview we consider also this benchmark in our comparison between fine-tuning-based and fine-tuning-free approaches, and report results in Tab. 6.5. Here the support set covers all the 1000 classes of ImageNet1k, which means that a fine-tuning is particularly expensive and avoiding it would preferable, as already observed in [111].

In general here the top-performing fine-tuning-based method is **Flow**. Still, if we look at the detailed results we notice that when the pre-training is on ImageNet21k Flow gets a higher average AUROC than the not-fine-tuned BiT CE prototypes, while the corresponding FPR95 values show an opposite trend. Moreover, when the pre-training dataset is LVD-142M, the best fine-tuned results produced by Flow are still worse than those of the not-fine-tuning DINOv2.

The best-performing method overall is once again DINOv2, which outperforms also MCM that used to hold the lead on this benchmark.

When looking at these results it is also important to keep in mind the difference in terms of computational cost between the fine-tuning-based methods and the fine-tuning-free ones. Indeed, with fine-tuning, we refer in practice to a *full-network* training session which may require a significant amount of time and has to be repeated for each OOD detection problem. This means that to obtain the numbers of a single row in Tab 6.4, fine-tuning-based methods perform 48 training sessions (considering the three random data orders for each experiment), while fine-tuning-free methods use the same fixed pre-trained model for all the experiments.

6.4.3 A *Wise* way to use fine-tuning

We have stated multiple times that the fine-tuning process may lead to *forgetting* part of the general knowledge that was originally present in the pre-trained model in favor of the new information collected from the fine-tuning data, a phenomenon that can be particularly problematic in the presence of a semantic shift between *support* data,

Table 6.6: Comparison between fine-tuning-based, fine-tuning-free models and WiSE-FT, using the KNN_OOD evaluator. We use the bold font to highlight the best result per column.

Pretraining Dataset	Backbone	Pretraining Objective	Fine-tuning	Intra-domain track												Cross-domain track			
				Textures		PatterNet		SUN		Stanford Cars		DomainNet Intra		AVG		SS → ST		MS → ST	
				AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓
ImageNet1k	ViT-B	CE	no	0.725	0.804	0.914	0.380	0.628	0.892	0.575	0.928	0.686	0.836	0.705	0.768	0.562	0.922	0.597	0.905
			yes	0.808	0.692	**0.992**	**0.043**	0.785	0.768	0.861	0.621	0.842	0.661	0.858	0.557	0.644	0.899	0.735	0.840
			WiSE-FT	0.783	0.742	0.982	0.091	0.711	0.821	0.644	0.897	0.801	0.713	0.784	0.653	0.625	0.901	0.711	0.839
		DINO	no	0.762	0.765	0.953	0.237	0.659	0.845	0.566	0.928	0.702	0.815	0.728	0.718	0.575	0.907	0.617	0.873
			yes	0.786	0.725	0.986	0.073	0.759	0.801	0.824	0.746	0.825	0.695	0.836	0.608	0.638	0.897	0.712	0.863
			WiSE-FT	0.780	0.728	0.983	0.091	0.719	0.811	0.709	0.898	0.797	0.723	0.797	0.650	0.624	0.895	0.703	0.842
ImageNet21k	BiT (Res101x3)	CE	no	0.832	0.635	0.938	0.303	0.760	0.775	0.582	0.923	0.724	0.766	0.767	0.680	0.589	0.899	0.643	0.845
			yes	0.797	0.730	0.986	0.077	0.789	0.765	0.857	0.676	0.842	0.653	0.854	0.580	0.640	0.900	0.721	0.843
			WiSE-FT	0.830	0.622	0.989	0.051	0.813	0.699	0.803	0.769	0.853	0.601	0.858	0.548	0.671	0.870	0.746	0.815
LVD-142M	ViT-L	DINOv2	no	0.790	0.759	0.912	0.386	0.772	0.744	0.716	0.852	0.789	0.685	0.796	0.685	0.678	0.835	0.711	0.812
			yes	0.840	0.616	0.986	0.060	0.822	0.703	**0.913**	**0.350**	0.867	0.574	0.886	0.461	0.696	0.847	0.748	0.832
			WiSE-FT	**0.846**	**0.610**	0.986	0.069	**0.846**	**0.606**	0.905	0.384	**0.874**	**0.517**	**0.891**	**0.437**	**0.738**	**0.794**	**0.786**	**0.751**

used for fine-tuning, and *test* data [74]. Recently, the authors of WiSE-FT [167] have suggested linearly combining pre-trained models with fine-tuned ones, showing that this process greatly increases the robustness to distribution shifts of the latter while retaining their performance on the fine-tuning data distribution. This study is particularly relevant to our whole analysis as we deal with distribution-shifted scenarios and till here we have performed a comparison between fine-tuning-free and fine-tuning-based methods treating them as irreconcilable approaches. We decide thus to test the proposed linear combination technique in our benchmark, both in the intra-domain and cross-domain tracks. While the original analysis of WiSE-FT specifically focused on CLIP-based models [129], which allow for a zero-shot straightforward classification by leveraging the text encoder, we instead consider the vision-only based models that produced the best results in Tab. 6.4. Specifically, we choose the KNN_norm evaluator for our comparison. Regarding the models' combination, we adopt equal weights (0.5), following the authors' recommendation. The obtained results are presented in Tab. 6.6 from which we can observe two different trends: on smaller models (*i.e.* , ViT-B CE and DINO) WiSE-FT obtains considerably worse performance compared to the sole fine-tuned networks, both in the intra-domain and in the cross-domain settings. Meanwhile, more complex models trained on larger datasets seem to benefit from the interpolation, with both BiT-CE and ViT-L-DINOv2 showing a significant advantage in the cross-domain scenario, while also slightly improving in the intra-domain one. These results seem to suggest that large models have enough representational capacity to benefit both from the general knowledge coming from the pre-training and from the detailed one acquired with the fine-tuning.

6.5 Conclusions

In this chapter, we have focused on a rather new research topic, carrying out experiments involving recent advancements in the field.

We started by discussing **two aspects** of the current research: on one side the **standard practice**, in the OOD detection literature, of training deep networks on the in-distribution data of the task at hand, a procedure that is considered necessary in order to learn the *normal* distribution and obtain good performance on the ID data, but that is inevitably connected to the risk of hurting the out-of-distribution representation capabilities. On the other side, we have described the **significant recent improvements** achieved in the literature studying representation learning on large data collections.

Starting from these considerations we have decided to carry out a **thorough analysis** on the potentialities of adopting OOD detection algorithms able to exploit **pre-trained representations**. We have presented the first representation learning paradigm designed explicitly to support OOD detection without fine-tuning. This algorithm acts as a ***proof-of-concept*** of the feasibility of the novel OOD detection paradigm, based on an explicit modeling of the *normality* and on direct comparisons between this normality and the *test* samples. With the aim to perform a large-scale experimental comparison between fine-tuning-free and fine-tuning-based OOD detection algorithms, we have designed a **novel benchmark**, after highlighting the disadvantages of the testbeds usually adopted in literature, which focus on very small datasets or consider only *far OOD* scenarios. Our novel benchmark consists of two tracks, the purpose of the intra-domain one is to enable studying specifically the ability of models to detect semantic shifts, while the cross-domain one is designed to reproduce a more realistic deployment scenario in which semantic and visual distribution shifts occur together. Our **experimental evaluation** of a wide range of algorithms on this novel benchmark allowed us to draw some conclusions about the performance of fine-tuning-free approaches w.r.t. fine-tuning-based ones. We have seen that the latter still keep the lead when there is no domain shift between *support* and *test* data, while their advantage resets to zero when the two distribution shift types occur, providing further proof of the reduced OOD representation capabilities of models fine-tuned on in-distribution data. The most relevant outcome of this chapter is however the empirical proof, for the first time, of the **superiority** of the ***foundation models*** w.r.t. traditional representation learning paradigms, in terms of direct applications of their representations to the OOD detection task. This finding fits perfectly into the context of the recent studies about the potentialities of *foundation models* which often highlight their advantages and thus focus on how to make the best use of the knowledge that they encode [191, 184].

Chapter 7

Conclusions and future opportunities

7.1 Summary

This thesis focused on the study of two significant issues that often arise when trying to move deep learning techniques from lab settings to real-world applications. Both issues appear in scenarios where it is not possible to control the data distribution met after deployment. Visual domain shifts occur when the visual distribution changes, because of different data acquisition conditions, while semantic shift happens when novel semantic categories are met after deployment. In our analysis, we have first of all described these problems, their causes, and their impacts on deep learning algorithms (Chap. 1), and then provided formal formulations for their most studied settings, and some background on the literature tackling them (Chap 2). In the subsequent chapters, the focus moved to more specific scenarios, for which we try to propose novel approaches, often adopting unconventional points of view.

In Chapter 3 we have analyzed the scenario encountered by a Computer Vision algorithm that is tasked to perform social media monitoring. In this research setting, which we call *one-shot unsupervised cross-domain detection*, traditional domain-adaptive detectors fail as they are not able to adapt to a continuously varying target domain. The problem should thus be tackled by performing adaptation on a single target sample at a time.

In Chapter 4 we carried out an attempt to recompose the Domain Generalization research field, which is split into two research lines apparently irreconcilable: on one side papers that study learning-level approaches to improve generalization, on the other, studies focusing on strategies to obtain generalization by increasing data variability. We find out that a very simple approach belonging to the second group can represent a novel strong baseline in the field, which should push for the development of novel algorithms able to make the most out of the enriched data.

Chapter 5 is devoted to the study of two different open-world learning settings,

both characterized by multiple challenges including the simultaneous presence of both a semantic and a visual distribution shift. The traditional paradigm applied here consists in building solutions obtained as combinations of methods designed for specific sub-problems. Our proposal is based on an opposite paradigm as we argue that, when dealing with a complex task, a dependable algorithm should tackle all its challenges at once.

Chapter 6 focuses on the Out-Of-Distribution detection task, proposing to discard the usual approach based on carrying out a training on the in-distribution data of the task at hand. Indeed, this procedure has a number of disadvantages and the recent improvements in terms of learning of general-purpose representations enable adopting a different approach based on the use of pre-trained feature representations. We analyze this novel direction, comparing it with the traditional one and we draw a comprehensive picture of the state-of-the-art in order to guide future research efforts.

7.2 Limitations and future opportunities

The knowledge presented in the thesis is the result of a long research journey that progressed hand-in-hand with the evolutions of the literature studying the distribution shift. For this reason, some of the insights provided and techniques proposed in the first chapters may appear **less relevant** with respect to the research outcomes of the last one. If we adopt a birds-eye view to look at the topics discussed in the thesis we notice that behind the innovations presented in each chapter, there has been a slow but steady change in perspective.

Initially (Chapters 3 and 4) we have focused on techniques to bridge the visual domain shift problem. They operate at the **learning protocol** level, putting more attention on the **training strategy** than on the **learned representations**. Moving forward (Chapter 5) we focused on the structure of the **learned feature space**. In this context, we observed how the choice of the learning objective impacts the learned features as well as the non-learned ones. Specifically, the joint effect of self-supervision and full-supervision in supervised contrastive learning provides models that generalize better. Our conclusions share insights with [36] that defined the notion of *supervision collapse*. Among the implications of this phenomenon, is the fact that **representations learned on a specific task and dataset are necessarily unsuited to be applied to out-of-distribution data**. This consideration prompted the studies that finally led to applying a broader perspective on distribution shift analysis (Chapter 6), by focusing on pre-trained representations as a way to provide fair treatment to in-distribution and out-of-distribution data.

Building on this result means transitioning from studying *how to train a model to solve a specific task*, to *how to query the knowledge encoded through a large-scale pre-training to solve the same task*. Indeed, while *foundation models* represent the latest innovation in terms of large-scale learning, exploiting them is not necessarily trivial,

and there is the serious risk of damaging the knowledge they encode when they are naively used as a starting point for transfer learning.

Considering this novel paradigm there are two main open lines of research: on one side the continuous improvement of pre-trained representations, which could be obtained through innovative learning objectives or the inclusion of ever-growing datasets or additional modalities, on the other side the design of specialized techniques to exploit those pre-trainings. The first point is today more and more under the control of very few entities that have access to the resources needed to undertake that large-scale effort, while the second aspect leaves more doors open and has already given birth to a new world of research directions including *prompt* learning, the introduction of *adapters*, and the use of *low-rank reparametrization* methods [88].

Some of these innovations may be suited to be applied to the distribution-shifted scenarios studied in this thesis, effectively obtaining approaches that can fully exploit foundation models, while still providing a way to adapt them to downstream tasks. If we consider for example cross-domain scenarios as the *one-shot unsupervised cross-domain detection* problem or the *multi-source open-set domain adaptation one*, a promising line of research would be to apply the adapters logic to the transformer-based architectures of foundation models. Indeed *adapters* were originally designed specifically for *multiple-domain learning* [131] but applied to traditional convolutional networks. An alternative strategy based on a similar rationale is the adoption of visual prompts [61]. Both approaches could be seen as strategies to enrich a frozen network with a small set of learnable parameters, that can be adapted to the target visual domain for example through self-supervised learning or entropy minimization [147]. An additional advantage of this paradigm w.r.t. full network fine-tuning is that the limited number of trainable parameters makes the adaptation much cheaper in terms of computational cost and size of data. Prompt learning can also be used for designing *OOD detection* techniques, for example through the introduction of negative prompts that enable measuring the dissimilarity with known classes [120].

While focusing on these novel research lines is certainly promising it is also important to always keep a critical eye on the whole framework, because even if the most recent findings seem to prove the superiority of pre-trained representations, as shown in the last sections of Chapter 6 there may always be situations in which the traditional approach of fine-tuning those representations provides better results.

Appendix A

Tasks and performance metrics

A.1 Object recognition

A.1.1 Task definition and goal

Object recognition, also known as *object classification* or *categorization*, is probably the most simple and common among the tasks studied in the Computer Vision research field. In its most common formulation, it consists in the task of correctly assigning an image to a predefined and closed set of *categories*, assuming that each image contains and represents a **single subject** that can be unambiguously assigned to a **single category**. In this context, a *classification model* is generally trained on a supervised dataset containing pairs of image samples and labels, where the latter simply identifies one of the known categories. At test time this *classifier* receives a test image and produces in output a scalar score for each category, which represents the model's computed probability that the image's subject belongs to the considered category. In most cases, the predicted class is the one with the highest associated probability, and the corresponding probability value is called *classification confidence*.

A.1.2 Performance metrics

The main performance metric used for categorization tasks is the so called *classification accuracy*:

$$accuracy = \frac{number\ of\ correctly\ classified\ test\ samples}{total\ number\ of\ test\ samples} \tag{A.1}$$

This metric is really simple and pretty common, given its significant interpretability. At the same time, it is important to keep in mind that its results may be misleading, especially in case of highly unbalanced classification tasks, for which other metrics may be preferred. One solution in this case consists in using the average per-class accuracy.

A.2 Visual object detection

A.2.1 Task definition and goal

Visual object detection consists in recognizing in an image all the objects that belong to a predefined set of categories, by also pointing out where they are by framing them in *bounding boxes*. This task can be seen as a specific formulation of a *set prediction* problem: each image contains a finite set of objects that belong to the known categories, and a detector should be able to correctly locate and classify all of them. The training dataset in this case contains image-label pairs where the label is a set of pairs (c, b), where c indicates a specific class and b the coordinates of a bounding box. A similar set of (c, b) pairs is the expected output of a good detector. In order to compute performance metrics, in many cases a predictive model also needs to provide a confidence score for each of its predictions, which is a scalar value with a similar meaning to the one of the object recognition's case.

A.2.2 Performance metrics

Differently from the classification case, evaluating visual object detection requires defining much more complex performance metrics, due to the complex structure of the task's expected outputs. A direct consequence of this is that there are different metrics that have been proposed by different entities and employed in different cases.

All the works included in this thesis dealing with the object detection task adopt benchmarks involving the Pascal-VOC dataset, which was introduced as part of the same named challenge [38]. As a result, we adopt the performance metrics defined as part of that challenge, the most important one being the *Average Precision* (AP), which is however obtained as a function of a number of other metrics. The AP is computed on a per-class basis and thus also the metrics on which it is based are computed following the same strategy. This means that when evaluating the performance of an object detector we consider a class at a time and evaluate the detector's ability to detect it. After considering all the classes we can provide a summary of the performance by computing the *mean Average precision* (**mAP**), which is simply the average of the per-class AP values.

Intersection over Union It is a measure of the overlap between 2 bounding boxes and is usually used to evaluate how well a proposed bounding box matches the corresponding ground truth (GT). It is computed as the ratio of the surface of the intersection between the two boxes and the surface of their union.

$$IoU = \frac{Intersection's\ surface}{Union's\ surface} \tag{A.2}$$

The IoU takes values in [0,1] and it is used together with a threshold to decide if a predicted bounding box matches a GT object or the background.

Precision-Recall curve The IoU allows to classify predicted and GT boxes as *True Positives* (**TP**, predicted box with IoU with a GT box higher than a threshold), *False Positives* (**FP**, IoU lower than the threshold, or GT box already matched with another prediction), and *False Negatives* (**FN**, a GT box not detected).
By counting instances of the various types it is possible to compute the *Precision* and the *Recall*:

$$Precision = \frac{TP}{TP + FP}; \qquad Recall = \frac{TP}{TP + FN}; \tag{A.3}$$

Both these metrics take values in [0,1]. Considering they are connected, with efforts of improving on one leading to a decrease in performance on the other, an overall performance evaluation can be performed by drawing the *Precision-Recall curve*, which shows the levels of precision for various levels of recall. This curve is obtained by varying the confidence threshold, *i.e.* the confidence value that the predictions have to reach in order to be included in the computation. The more the curve resembles the *Heaviside step function* the more the detector's performance is good.

Average Precision Because the Precision-Recall curve does not allow to easily compare the performance of two different models, a numeric value is usually preferred. This is obtained by measuring the area under the curve, whose value in [0,1] is called *Average Precision* (AP). The AP performance results reported in this thesis are computed with an IoU threshold of 0.5.

A.3 Out-Of-Distribution detection

A.3.1 Task definition and goal

Out-Of-Distribution detection is a *categorization* problem that involves a semantic shift. In this case, the task consists in associating each test image either to the *known* categories or to the *unknown* one. The task can thus be seen as a *binary classification* problem on the {*known*, *unknown*} class set, where a full prediction output can be provided as a single scalar value often called *normality score*, which represents the prediction's *confidence* for the known class. The main difference and peculiarity of Out-Of-Distribution detection with respect to many other binary classification problems is that in its most used formulation, the training dataset contains samples of only one of the two classes involved in the task (the *known* one), while the other class is not clearly defined and simply corresponds to the set of all semantics that do not appear in the known set.

A.3.2 Performance metrics

Given the fact that Out-Of-Distribution detection can be considered a binary classification task, the performance metrics usually employed in papers studying this task are

based on the concepts of **True Positive** (TP, known samples detected as known), **False Positive** (FP, unknown samples detected as known), **True Negative** (TN, unknown samples detected as unknown) and **False Negative** (FN, known samples detected as unknown).

AUROC It is the Area Under the Receiver Operating Characteristic (ROC) curve, which plots the True Positive Rate (TPR = TP/TP + FN) against the False Positive Rate (FPR = FP/FP + TN) when varying the decision threshold applied on the normality scores. This metric takes value in [0,1] (the higher the better) and can be interpreted as the probability that a *known* test sample has to receive a higher normality score than an *unknown* one. One of the main advantages of this metric is that it is threshold-free so it provides an unbiased evaluation of the ability of an Out-Of-Distribution detection method to separate known and unknown samples.

FPR95 It represents the False Positive Rate when the True Positive Rate is 95%, for this reason, it is sometimes indicated ad FPR@TPRx with $x = 95\%$. As the AUROC, it takes values in [0,1], but in this case, a lower value is better. Moreover, this is not a threshold-free metric, but it is based on the choice of the normality threshold that guarantees that 95% of positive samples are predicted as positives.

A.4 Open-set domain adaptation

A.4.1 Task definition and goal

Open-set domain adaptation is an *object recognition* problem that considers both a visual domain and a semantic shift. The classification model in this case is trained both on a supervised *source* dataset and on an unsupervised *target* one. At test time the classifier receives a single target sample at a time and must provide for it a classification prediction over the $\{known\ classes\} \bigcup \{unknown\}$ class set.

A.4.2 Performance metrics

Algorithms designed for the open-set domain adaptation task are usually evaluated considering their ability to both classify known samples correctly while detecting unknown samples. We indicate with $\mathbf{OS}^*$ the average class accuracy over known classes and with **UNK** the classification accuracy for the unknown class (considered as a whole as if it was a single uniform semantic class).

The traditional [93, 75] approach to aggregate these two metrics consists in computing an overall average of per-class accuracies:

$$\mathbf{OS} = \frac{|\mathcal{Y}_S|}{|\mathcal{Y}_S| + 1} \cdot \mathbf{OS}^* + \frac{1}{|\mathcal{Y}_S| + 1} \cdot \mathbf{UNK}$$

where $|\mathcal{Y}_S|$ represents the cardinality of the known class set. This metric however gives all the classes the same importance and consider the unknown one not different from the others. As a result it is possible to obtain a really high **OS** even with a low **UNK** if the known class accuracy is high and $|\mathcal{Y}_S|$ is also high.

A more fair aggregation metric has been proposed in [14, 42]: it is the harmonic mean between the average class accuracy over the known classes $\mathbf{OS}^*$ and the accuracy over the unknown class **UNK**:

$$\mathbf{HOS} = 2 \cdot \frac{\mathbf{OS}^* \cdot \mathbf{UNK}}{\mathbf{OS}^* + \mathbf{UNK}} \tag{A.4}$$

With this metric any method that shows good classification ability only towards one among known and unknown samples cannot obtain a good overall score.

A.5 Open-world recognition

A.5.1 Task definition and goal

Open-world recognition (OWR) is a categorization task based on an incremental learning paradigm. After each learning episode, an OWR model should be able to correctly recognize all the test samples belonging to the classes learned till that point, while rejecting all the samples belonging to other semantic categories. Similarly to open-set domain adaptation, and differently w.r.t Out-Of-Distribution detection, this task requires a model not only to provide a normality score for each test sample but also to propose a known-unknown separation threshold. The model is thus evaluated both on its ability to correctly classify known samples and to correctly detect unknown ones.

A.5.2 Performance metrics

The performance metrics that we adopt for this task are the one proposed in [41]:

- **Acc** (Accuracy) measures the ability of the model to correctly classify the known target samples.
- **Acc-WR** (Accuracy Without Rejection) is similar to Acc, but the accuracy is computed without rejecting the target samples identified as unknown. In practice, this metric is computed considering all known test samples as known, even if they have a normality score lower than the normality threshold. As a result, w.r.t to *Acc.*, it focuses only on the closed set classification accuracy of the model;
- **OWR-H** (Open World Harmonic Mean) evaluates the performance of the model as a whole, it is the harmonic mean between Acc-WR and the model's accuracy in unknown sample detection. Similarly to the *HOS* score of open-set domain

adaptation task, it is the only metric that summarizes the overall performance of the model in a single value and thus the most important to be used when comparing different approaches.

Bibliography

[1] Davide Abati et al. "Latent Space Autoregression for Novelty Detection." In: *CVPR*. 2019.

[2] Hongjoon Ahn et al. "SS-IL: Separated Softmax for Incremental Learning." In: *ICCV*. 2021.

[3] Antonio Alliegro, Francesco Cappio Borlino, and Tatiana Tommasi. "3DOS: Towards 3D Open Set Learning - Benchmarking and Understanding Semantic Novelty Detection on Point Clouds." In: *NeurIPS Datasets and Benchmarks Track*. 2022.

[4] Shai Ben-David et al. "A theory of learning from different domains." In: *Machine Learning* 79 (2010), pp. 151–175. URL: http://www.springerlink.com/content/q6qk230685577n52/.

[5] Sagie Benaim and Lior Wolf. "One-Shot Unsupervised Cross Domain Translation." In: *NIPS*. 2018.

[6] Abhijit Bendale and Terrance Boult. "Towards open world recognition." In: *CVPR*. 2015.

[7] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc., 2009.

[8] Daniel Bolya et al. "TIDE: A General Toolbox for Identifying Object Detection Errors." In: *ECCV*. 2020.

[9] Rishi Bommasani et al. "On the opportunities and risks of foundation models." In: *arXiv preprint arXiv:2108.07258* (2021).

[10] Florian Bordes et al. "Guillotine Regularization: Why removing layers is needed to improve generalization in Self-Supervised Learning." In: *Transactions on Machine Learning Research* (2023).

[11] Karsten M. Borgwardt et al. "Integrating structured biological data by Kernel Maximum Mean Discrepancy." In: *Bioinformatics* 22.14 (July 2006), e49–e57. DOI: 10.1093/bioinformatics/btl242.

[12] Konstantinos Bousmalis et al. "Unsupervised Pixel-Level Domain Adaptation With Generative Adversarial Networks." In: *CVPR*. 2017.

[13] Silvia Bucci, Antonio D'Innocente, and Tatiana Tommasi. "Tackling Partial Domain Adaptation with Self-Supervision." In: *ICIAP*. 2019.

[14] Silvia Bucci, Mohammad Reza Loghmani, and Tatiana Tommasi. "On the Effectiveness of Image Rotation for Open Set Domain Adaptation." In: *ECCV*. 2020.

[15] Silvia Bucci et al. "Distance-Based Hyperspherical Classification for Multi-Source Open-Set Domain Adaptation." In: *WACV*. 2022.

[16] Silvia Bucci et al. "Self-Supervised Learning Across Domains." In: *IEEE TPAMI* 44.9 (2022), pp. 5516–5528.

[17] Francesco Cappio Borlino, Silvia Bucci, and Tatiana Tommasi. "Contrastive Learning for Cross-Domain Open World Recognition." In: *IROS*. 2022.

[18] Francesco Cappio Borlino, Silvia Bucci, and Tatiana Tommasi. "Semantic Novelty Detection via Relational Reasoning." In: *ECCV*. 2022.

[19] Francesco Cappio Borlino, Antonio D'Innocente, and Tatiana Tommasi. "Rethinking Domain Generalization Baselines." In: *ICPR*. 2021.

[20] Francesco Cappio Borlino, Lorenzo Li Lu, and Tatiana Tommasi. "Foundation Models and Fine-Tuning: A Benchmark for Out Of Distribution Detection." In: *IEEE Access* (2024).

[21] Francesco Cappio Borlino et al. "Self-supervision & meta-learning for one-shot unsupervised cross-domain detection." In: *CVIU* 223.C (2022).

[22] Nicolas Carion et al. "End-to-End Object Detection with Transformers." In: *ECCV*. 2020.

[23] Fabio Maria Carlucci et al. "Domain Generalization by Solving Jigsaw Puzzles." In: *CVPR*. 2019.

[24] Mathilde Caron et al. "Emerging Properties in Self-Supervised Vision Transformers." In: *ICCV*. 2021.

[25] Guangyao Chen et al. "Adversarial Reciprocal Points Learning for Open Set Recognition." In: *IEEE TPAMI* (2021).

[26] Ting Chen et al. "A Simple Framework for Contrastive Learning of Visual Representations." In: *ICML*. 2020.

[27] Yuhua Chen et al. "Domain Adaptive Faster R-CNN for Object Detection in the Wild." In: *CVPR*. 2018.

[28] Ying Cheng et al. "Look, Listen, and Attend: Co-Attention Network for Self-Supervised Audio-Visual Representation Learning." In: *ACM Multimedia*. 2020.

[29] Mircea Cimpoi et al. "Describing Textures in the Wild." In: *CVPR*. 2014.

[30] Tomer Cohen and Lior Wolf. "Bidirectional One-Shot Unsupervised Domain Mapping." In: *ICCV*. 2019.

[31] Marius Cordts et al. "The cityscapes dataset for semantic urban scene understanding." In: *CVPR*. 2016.

[32] Antonio D'Innocente and Barbara Caputo. "Domain Generalization with Domain-Specific Aggregation Modules." In: *GCPR*. 2018.

[33] Antonio D'Innocente et al. "One-Shot Unsupervised Cross-Domain Detection." In: *ECCV*. 2020.

[34] Jia Deng et al. "Imagenet: A large-scale hierarchical image database." In: *CVPR*. 2009.

[35] Andrija Djurisic et al. "Extremely Simple Activation Shaping for Out-of-Distribution Detection." In: *ICLR*. 2023.

[36] Carl Doersch, Ankush Gupta, and Andrew Zisserman. "CrossTransformers: spatially-aware few-shot transfer." In: *NeurIPS*. 2020.

[37] Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale." In: *ICLR*. 2021.

[38] Mark Everingham et al. "The pascal visual object classes (voc) challenge." In: *IJCV* 88.2 (2010), pp. 303–338.

[39] Chelsea Finn, Pieter Abbeel, and Sergey Levine. "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks." In: *ICML*. 2017.

[40] Dario Fontanel et al. "Boosting deep open world recognition by clustering." In: *IEEE RAL* 5.4 (2020).

[41] Dario Fontanel et al. "On the Challenges of Open World Recognition Under Shifting Visual Domains." In: *IEEE RAL* 6.2 (2021).

[42] Bo Fu et al. "Learning to Detect Open Classes for Universal Domain Adaptation." In: *ECCV*. 2020.

[43] Yaroslav Ganin et al. "Domain-adversarial Training of Neural Networks." In: *JMLR* 17.1 (2016), pp. 2096–2030.

[44] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. "Unsupervised Representation Learning by Predicting Image Rotations." In: *ICLR*. 2018.

[45] Boqing Gong, Kristen Grauman, and Fei Sha. "Learning Kernels for Unsupervised Domain Adaptation with Applications to Visual Object Recognition." In: *Int. J. Comput. Vision* 109 (2014), pp. 3–27. DOI: 10.1007/s11263-014-0718-4.

[46] Ian Goodfellow et al. "Generative Adversarial Nets." In: *NeurIPS*. 2014.

[47] Ishaan Gulrajani and David Lopez-Paz. "In Search of Lost Domain Generalization." In: *ICLR*. 2021.

[48] Kaiming He et al. “Deep Residual Learning for Image Recognition.” In: *CVPR*. 2016.

[49] Kaiming He et al. “Momentum Contrast for Unsupervised Visual Representation Learning.” In: *CVPR*. 2020.

[50] Dan Hendrycks and Kevin Gimpel. “A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks.” In: *ICLR*. 2017.

[51] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. “Deep Anomaly Detection with Outlier Exposure.” In: *ICLR* (2019).

[52] Judy Hoffman et al. “CyCADA: Cycle-Consistent Adversarial Domain Adaptation.” In: *ICML*. 2018.

[53] Rui Huang, Andrew Geng, and Yixuan Li. “On the Importance of Gradients for Detecting Distributional Shifts in the Wild.” In: *NeurIPS*. 2021.

[54] Rui Huang and Yixuan Li. “MOS: Towards Scaling Out-of-Distribution Detection for Large Semantic Space.” In: *CVPR*. 2021.

[55] Xun Huang and Serge Belongie. “Arbitrary Style Transfer in Real-Time With Adaptive Instance Normalization.” In: *ICCV*. 2017.

[56] Zeyi Huang et al. “Self-Challenging Improves Cross-Domain Generalization.” In: *ECCV*. 2020.

[57] Naoto Inoue et al. “Cross-Domain Weakly-Supervised Object Detection Through Progressive Domain Adaptation.” In: *CVPR*. 2018.

[58] Simon Jenni, Hailin Jin, and Paolo Favaro. “Steering Self-Supervised Feature Learning Beyond Local Pixel Statistics.” In: *CVPR*. 2020.

[59] Rae Jeong et al. “Self-Supervised Sim-to-Real Adaptation for Visual Robotic Manipulation.” In: *ICRA*. 2020.

[60] Chao Jia et al. “Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision.” In: *ICML*. 2021.

[61] Menglin Jia et al. “Visual Prompt Tuning.” In: *European Conference on Computer Vision (ECCV)*. 2022.

[62] Ying Jin et al. “Minimum Class Confusion for Versatile Domain Adaptation.” In: *ECCV*. 2020.

[63] Justin Johnson et al. “CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning.” In: *CVPR*. 2017.

[64] Tejaswi Kasarla et al. “Maximum Class Separation as Inductive Bias in One Matrix.” In: *NeurIPS*. 2022.

[65] Mehran Khodabandeh et al. “A Robust Learning Approach to Domain Adaptive Object Detection.” In: *ICCV*. 2019.

[66] Prannay Khosla et al. "Supervised Contrastive Learning." In: *NeurIPS*. 2020.

[67] Seunghyeon Kim et al. "Self-Training and Adversarial Background Regularization for Unsupervised Domain Adaptive One-Stage Object Detection." In: *ICCV*. 2019.

[68] Taekyung Kim et al. "Diversify and Match: A Domain Adaptive Representation Learning Paradigm for Object Detection." In: *CVPR*. 2019.

[69] Ikki Kishida et al. "Object Recognition With Continual Open Set Domain Adaptation for Home Robot." In: *WACV*. 2021.

[70] Alexander Kolesnikov et al. "Big Transfer (BiT): General Visual Representation Learning." In: *ECCV*. 2020.

[71] Simon Kornblith et al. "Why Do Better Loss Functions Lead to Less Transferable Features?" In: *NeurIPS*. 2021.

[72] Jonathan Krause et al. "3D Object Representations for Fine-Grained Categorization." In: *ICCV Workshops*. 2013.

[73] Alex Krizhevsky and Geoffrey Hinton. *Learning multiple layers of features from tiny images*. Tech. rep. 2009. URL: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[74] Ananya Kumar et al. "Fine-Tuning can Distort Pretrained Features and Underperform Out-of-Distribution." In: *ICLR*. 2022.

[75] Jogendra Nath Kundu et al. "Towards inheritable models for open-set domain adaptation." In: *CVPR*. 2020.

[76] Kevin Lai et al. "A large-scale hierarchical multi-view rgb-d object dataset." In: *ICRA*. 2011.

[77] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. "Simple and scalable predictive uncertainty estimation using deep ensembles." In: *NeurIPS*. 2017.

[78] Yann LeCun, Corinna Cortes, and CJ Burges. *Mnist hand- written digit database*. Tech. rep. 2010. URL: http://yann.lecun.com/exdb/mnist.

[79] Kimin Lee et al. "A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks." In: *NeurIPS*. 2018.

[80] Kimin Lee et al. "Training confidence-calibrated classifiers for detecting out-of-distribution samples." In: *ICLR*. 2017.

[81] Da Li et al. "Deeper, Broader and Artier Domain Generalization." In: *ICCV*. 2017.

[82] Da Li et al. "Episodic training for domain generalization." In: *ICCV*. 2019.

[83] Da Li et al. "Learning to Generalize: Meta-Learning for Domain Generalization." In: *AAAI*. 2018.

[84] Haoliang Li et al. "Domain Generalization With Adversarial Feature Learning." In: *CVPR*. 2018.

[85] Xinhao Li et al. "Interpretable Open-Set Domain Adaptation via Angular Margin Separation." In: *ECCV*. 2022.

[86] Ya Li et al. "Deep Domain Generalization via Conditional Invariant Adversarial Networks." In: *ECCV*. 2018.

[87] Zhizhong Li and Derek Hoiem. "Learning without forgetting." In: *IEEE T-PAMI* 40.12 (2017).

[88] Vladislav Lialin, Vijeta Deshpande, and Anna Rumshisky. "Scaling Down to Scale Up: A Guide to Parameter-Efficient Fine-Tuning." In: *arXiv preprint arXiv:2303.15647* (2023).

[89] Jian Liang, Ran He, and Tieniu Tan. "A Comprehensive Survey on Test-Time Adaptation under Distribution Shifts." In: *arXiv preprint arXiv:2303.15361* (2023).

[90] Shiyu Liang, Yixuan Li, and R. Srikant. "Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks." In: *ICLR*. 2018.

[91] Tsung-Yi Lin et al. "Focal Loss for Dense Object Detection." In: *ICCV*. 2017.

[92] Tsung-Yi Lin et al. "Microsoft COCO: Common Objects in Context." In: *ECCV*. 2014.

[93] Hong Liu et al. "Separate to Adapt: Open Set Domain Adaptation via Progressive Separation." In: *CVPR*. 2019.

[94] Wei Liu et al. "Ssd: Single shot multibox detector." In: *ECCV*. 2016.

[95] Weitang Liu et al. "Energy-based Out-of-distribution Detection." In: *NeurIPS*. 2020.

[96] Mohammad Reza Loghmani, Barbara Caputo, and Markus Vincze. "Recognizing objects in-the-wild: Where do we stand?" In: *ICRA*. 2018.

[97] Mohammad Reza Loghmani et al. "Unsupervised domain adaptation through inter-modal rotation for rgb-d object recognition." In: *IEEE RAL* 5.4 (2020).

[98] Vincenzo Lomonaco and Davide Maltoni. "CORe50: a New Dataset and Benchmark for Continuous Object Recognition." In: *PMLR*. 2017.

[99] Mingsheng Long et al. "Conditional Adversarial Domain Adaptation." In: *NeurIPS*. 2018.

[100] Mingsheng Long et al. "Learning Transferable Features with Deep Adaptation Networks." In: *ICML*. 2015.

[101] Lorenzo Li Lu et al. "Large Class Separation is not what you need for Relational Reasoning-based OOD Detection." In: *ICIAP*. 2023.

[102] Yadan Luo et al. “Progressive Graph Learning for Open-Set Domain Adaptation.” In: *ICML*. 2020.

[103] Zelun Luo et al. “Label Efficient Learning of Transferable Representations acrosss Domains and Tasks.” In: *NeurIPS*. 2017.

[104] Zheda Mai et al. “Supervised Contrastive Replay: Revisiting the Nearest Class Mean Classifier in Online Class-Incremental Continual Learning.” In: *CVPRW*. 2021.

[105] Massimiliano Mancini et al. “Knowledge is never enough: Towards web aided deep open world recognition.” In: *ICRA*. 2019.

[106] Toshihiko Matsuura and Tatsuya Harada. “Domain Generalization Using a Mixture of Multiple Latent Domains.” In: *AAAI*. 2020.

[107] Thomas Mensink et al. “Metric Learning for Large Scale Image Classification: Generalizing to New Classes at Near-Zero Cost.” In: *ECCV*. 2012.

[108] Pascal Mettes, Elise van der Pol, and Cees Snoek. “Hyperspherical Prototype Networks.” In: *NeurIPS*. 2019.

[109] Benjamin J. Meyer and Tom Drummond. “The Importance of Metric Learning for Robotic Vision: Open Set Recognition and Active Learning.” In: *ICRA*. 2019.

[110] Yifei Ming and Yixuan Li. “How Does Fine-Tuning Impact Out-of-Distribution Detection for Vision-Language Models?” In: *International Journal of Computer Vision* 132.2 (2024), pp. 596–609. DOI: 10.1007/s11263-023-01895-7.

[111] Yifei Ming et al. “Delving into Out-of-Distribution Detection with Vision-Language Representations.” In: *NeurIPS*. 2022.

[112] Yifei Ming et al. “How to Exploit Hyperspherical Embeddings for Out-of-Distribution Detection?” In: *ICLR*. 2022.

[113] Saeid Motiian et al. “Few-Shot Adversarial Domain Adaptation.” In: *NIPS*. 2017.

[114] Saeid Motiian et al. “Unified Deep Supervised Domain Adaptation and Generalization.” In: *ICCV*. 2017.

[115] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. “Domain Generalization via Invariant Feature Representation.” In: *ICML*. 2013.

[116] Jishnu Mukhoti et al. “Calibrating Deep Neural Networks using Focal Loss.” In: *NeurIPS*. 2020.

[117] Hyeonseob Nam et al. “Reducing domain gap by reducing style bias.” In: *CVPR*. 2021.

[118] Anh Nguyen, Jason Yosinski, and Jeff Clune. “Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images.” In: *CVPR*. 2015.

[119] K. Nichol. *Painter by numbers, WikiArt*. 2016. URL: https://www.kaggle.com/c/painter-by-numbers.

[120] Jun Nie et al. "Out-of-Distribution Detection with Negative Prompts." In: *ICLR*. 2024.

[121] Mehdi Noroozi and Paolo Favaro. "Unsupervised learning of visual representations by solving jigsaw puzzles." In: *ECCV*. 2016.

[122] Maxime Oquab et al. "DINOv2: Learning Robust Visual Features without Supervision." In: *arXiv preprint arXiv:2304.07193* (2023).

[123] Pau Panareda Busto and Juergen Gall. "Open Set Domain Adaptation." In: *ICCV*. 2017.

[124] Dim P. Papadopoulos et al. "Extreme Clicking for Efficient Object Annotation." In: *ICCV*. 2017.

[125] Massimiliano Patacchiola and Amos Storkey. "Self-Supervised Relational Reasoning for Representation Learning." In: *NeurIPS*. 2020.

[126] Xingchao Peng et al. "Domain Agnostic Learning with Disentangled Representations." In: *ICML*. 2019.

[127] Xingchao Peng et al. "Moment Matching for Multi-Source Domain Adaptation." In: *ICCV*. 2019.

[128] Siyuan Qiao et al. "Micro-Batch Training with Batch-Channel Normalization and Weight Standardization." In: *ArXiv:1903.10520* (2019).

[129] Alec Radford et al. "Learning Transferable Visual Models From Natural Language Supervision." In: *ICML*. 2021.

[130] Sayan Rakshit et al. "Multi-source Open-Set Deep Adversarial Domain Adaptation." In: *ECCV*. 2020.

[131] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. "Learning multiple visual domains with residual adapters." In: *NeurIPS*. 2017.

[132] Sylvestre-Alvise Rebuffi et al. "iCaRL: Incremental Classifier and Representation Learning." In: *CVPR*. 2017.

[133] Joseph Redmon et al. "You only look once: Unified, real-time object detection." In: *CVPR*. 2016.

[134] Shaoqing Ren et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." In: *NIPS*. 2015.

[135] Paolo Russo et al. "From Source to Target and Back: Symmetric Bi-Directional Adaptive GAN." In: *CVPR*. 2018.

[136] Kate Saenko et al. "Adapting visual category models to new domains." In: *ECCV*. 2010.

[137] Kuniaki Saito et al. "Maximum Classifier Discrepancy for Unsupervised Domain Adaptation." In: *CVPR*. 2018.

[138] Kuniaki Saito et al. "Strong-Weak Distribution Alignment for Adaptive Object Detection." In: *CVPR*. 2019.

[139] Kuniaki Saito et al. "Universal Domain Adaptation through Self-Supervision." In: *NeurIPS*. 2020.

[140] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. "Semantic foggy scene understanding with synthetic data." In: *IJCV* 126.9 (2018), pp. 973–992.

[141] Adam Santoro et al. "A simple neural network module for relational reasoning." In: *NeurIPS*. 2017.

[142] Chandramouli Shama Sastry and Sageev Oore. "Detecting Out-of-Distribution Examples with Gram Matrices." In: *ICML*. 2020.

[143] Ramprasaath R. Selvaraju et al. "Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization." In: *ICCV*. 2017.

[144] Yang Shu et al. "Open Domain Generalization with Domain-Augmented Meta-Learning." In: *CVPR*. 2021.

[145] Hao Su, Jia Deng, and Li Fei-Fei. "Crowdsourcing annotations for visual object detection." In: *AAAI Human Computation Workshop*. 2012.

[146] Baochen Sun and Kate Saenko. "Deep CORAL: Correlation Alignment for Deep Domain Adaptation." In: *ECCV 2016 Workshops*. 2016.

[147] Jiachen Sun et al. "VPA: Fully Test-Time Visual Prompt Adaptation." In: *ACM MM*. 2023. ISBN: 9798400701085. DOI: 10.1145/3581783.3611835.

[148] Yiyou Sun, Chuan Guo, and Yixuan Li. "ReAct: Out-of-distribution Detection With Rectified Activations." In: *NeurIPS*. 2021.

[149] Yiyou Sun et al. "Out-of-Distribution Detection with Deep Nearest Neighbors." In: *ICML*. 2022.

[150] Yu Sun et al. "Test-Time Training with Self-Supervision for Generalization under Distribution Shifts." In: *ICML*. 2020.

[151] Flood Sung et al. "Learning to Compare: Relation Network for Few-Shot Learning." In: *CVPR*. 2018.

[152] Jihoon Tack et al. "CSI: Novelty Detection via Contrastive Learning on Distributionally Shifted Instances." In: *NeurIPS*. 2020.

[153] Joshua Tobin et al. "Domain randomization for transferring deep neural networks from simulation to the real world." In: *IROS*. 2017.

[154] Antonio Torralba and Alexei A. Efros. "Unbiased look at dataset bias." In: *CVPR*. 2011.

[155] Eric Tzeng et al. “Adversarial Discriminative Domain Adaptation.” In: *CVPR*. 2017.

[156] Grant Van Horn et al. “The INaturalist Species Classification and Detection Dataset.” In: *CVPR*. 2018.

[157] Ashish Vaswani et al. “Attention is All you Need.” In: *NeurIPS*. 2017.

[158] Sagar Vaze et al. “Open-Set Recognition: A Good Closed-Set Classifier is All You Need.” In: *ICLR*. 2022.

[159] Hemanth Venkateswara et al. “Deep Hashing Network for Unsupervised Domain Adaptation.” In: *CVPR*. 2017.

[160] Paul Viola and Michael Jones. “Rapid object detection using a boosted cascade of simple features.” In: *CVPR*. 2001.

[161] Riccardo Volpi and Vittorio Murino. “Addressing model vulnerability to distributional shifts over image transformation sets.” In: *ICCV*. 2019.

[162] Riccardo Volpi et al. “Generalizing to Unseen Domains via Adversarial Data Augmentation.” In: *NeurIPS*. 2018.

[163] Hao Wang et al. “CosFace: Large Margin Cosine Loss for Deep Face Recognition.” In: *CVPR*. 2018.

[164] Hualiang Wang et al. “CLIPN for Zero-Shot OOD Detection: Teaching CLIP to Say No.” In: *ICCV*. 2023.

[165] Tongzhou Wang and Phillip Isola. “Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere.” In: *ICML*. 2020.

[166] Hongxin Wei et al. “Mitigating Neural Network Overconfidence with Logit Normalization.” In: *ICML*. 2022.

[167] Mitchell Wortsman et al. “Robust Fine-Tuning of Zero-Shot Models.” In: *CVPR*. 2022.

[168] Aming Wu et al. “Vector-Decomposed Disentanglement for Domain-Invariant Object Detection.” In: *ICCV*. 2021.

[169] Yuxin Wu and Kaiming He. “Group Normalization.” In: *ECCV*. 2018.

[170] Jianxiong Xiao et al. “SUN database: Large-scale scene recognition from abbey to zoo.” In: *CVPR*. 2010.

[171] Chang-Dong Xu et al. “Exploring Categorical Regularization for Domain Adaptive Object Detection.” In: *CVPR*. 2020.

[172] J. Xu, L. Xiao, and A. M. López. “Self-Supervised Domain Adaptation for Computer Vision Tasks.” In: *IEEE Access* 7 (2019).

[173] Jiaolong Xu, Liang Xiao, and Antonio M. López. "Self-Supervised Domain Adaptation for Computer Vision Tasks." In: *IEEE Access* 7 (2019), pp. 156694–156706. DOI: 10.1109/ACCESS.2019.2949697.

[174] Minghao Xu et al. "Adversarial Domain Adaptation with Domain Mixup." In: *AAAI*. 2020.

[175] Ruijia Xu et al. "Larger Norm More Transferable: An Adaptive Feature Norm Approach for Unsupervised Domain Adaptation." In: *ICCV*. 2019.

[176] Jingkang Yang, Kaiyang Zhou, and Ziwei Liu. "Full-Spectrum Out-of-Distribution Detection." In: *IJCV* 131 (2023).

[177] Jingkang Yang et al. "Generalized Out-of-Distribution Detection: A Survey." In: *arXiv preprint arXiv:2110.11334* (2021).

[178] Yang You, Igor Gitman, and Boris Ginsburg. "Large batch training of convolutional networks." In: *arXiv preprint arXiv:1708.03888* (2017).

[179] Qing Yu and Kiyoharu Aizawa. "Unsupervised Out-of-Distribution Detection by Maximum Classifier Discrepancy." In: *ICCV*. 2019.

[180] Hongjie Zhang et al. "Hybrid Models for Open Set Recognition." In: *ECCV*. 2020.

[181] Hongyi Zhang et al. "mixup: Beyond Empirical Risk Minimization." In: *ICLR*. 2018.

[182] Jingyang Zhang et al. "Mixture Outlier Exposure: Towards Out-of-Distribution Detection in Fine-Grained Environments." In: *WACV*. 2023.

[183] Jingyang Zhang et al. "OpenOOD v1.5: Enhanced Benchmark for Out-of-Distribution Detection." In: *arXiv preprint arXiv:2306.09301* (2023).

[184] Renrui Zhang et al. "Prompt, Generate, Then Cache: Cascade of Foundation Models Makes Strong Few-Shot Learners." In: *CVPR*. 2023.

[185] Y. Zhang et al. "Learning Robust Shape-Based Features for Domain Generalization." In: *IEEE Access* 8 (2020).

[186] Yifan Zhang et al. "Unleashing the Power of Contrastive Self-Supervised Visual Models via Contrast-Regularized Fine-Tuning." In: *arXiv preprint arXiv:2102.06605* (2021).

[187] Bolei Zhou et al. "Places: A 10 million Image Database for Scene Recognition." In: *IEEE TPAMI* (2017).

[188] Kaiyang Zhou et al. "Conditional Prompt Learning for Vision-Language Models." In: *CVPR*. 2022.

[189] Kaiyang Zhou et al. "Deep Domain-Adversarial Image Generation for Domain Generalisation." In: *AAAI* (2020).

[190] Kaiyang Zhou et al. "Domain Generalization: A Survey." In: *IEEE TPAMI* (2023).

[191] Kaiyang Zhou et al. “Learning to Prompt for Vision-Language Models.” In: *International Journal of Computer Vision (IJCV)* (2022).

[192] Weixun Zhou et al. “PatternNet: A benchmark dataset for performance evaluation of remote sensing image retrieval.” In: *ISPRS Journal of Photogrammetry and Remote Sensing* (2018).

[193] Jun-Yan Zhu et al. “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks.” In: *ICCV*. 2017.

This Ph.D. thesis has been typeset by means of the TEX-system facilities. The typesetting engine was LuaLATEX. The document class was `toptesi`, by Claudio Beccari, with option `tipotesi=scudo`. This class is available in every up-to-date and complete TEX-system installation.